Java AWT Tutorial

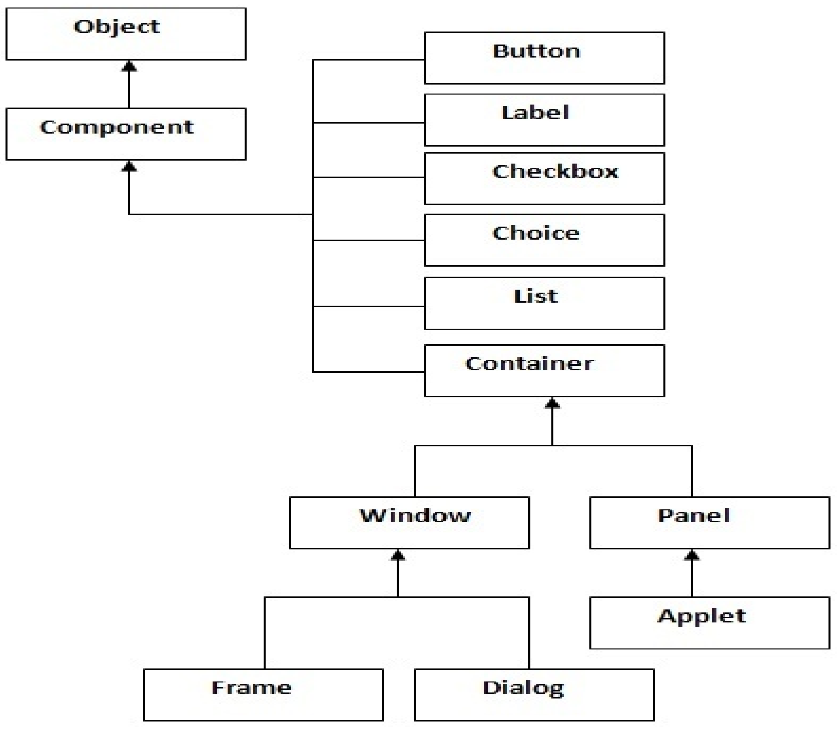
Java AWT (Abstract Windowing Toolkit) is *an API to develop GUI or window-based application in java*.

Java AWT components are platform-dependent i.e. components are displayed according to the view of operating system. AWT is heavyweight i.e. its components uses the resources of system.

The java.awt package provides classes for AWT api such as TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.

Java AWT Hierarchy

The hierarchy of Java AWT classes are given below.



Container

The Container is a component in AWT that can contain another components like buttons, textfields, labels etc. The classes that extends Container class are known as container such as Frame, Dialog and Panel.

Window

The window is the container that have no borders and menu bars. You must use frame, dialog or another window for creating a window.

Panel

The Panel is the container that doesn't contain title bar and menu bars. It can have other components like button, textfield etc.

Frame

The Frame is the container that contain title bar and can have menu bars. It can have other components like button, textfield etc.

Useful Methods of Component class

Method	Description
public void add(Component c)	inserts a component on this component.
public void setSize(int width,int height)	sets the size (width and height) of the component.
public void setLayout(LayoutManager m)	defines the layout manager for the component.
public void setVisible(boolean status)	changes the visibility of the component, by default false.

Java AWT Example

To create simple awt example, you need a frame. There are two ways to create a frame in AWT.

- By extending Frame class (inheritance)
- By creating the object of Frame class (association)

Simple example of AWT by inheritance

```
import java.awt.*;
class First extends Frame{
First(){
Button b=new Button("click me");
b.setBounds(30,100,80,30);// setting button position

add(b);//adding button into frame
setSize(300,300);//frame size 300 width and 300 height
setLayout(null);//no layout manager
setVisible(true);//now frame will be visible, by default not visible
}
public static void main(String args[]){
First f=new First();
}}
```

Simple example of AWT by association

```
import java.awt.*;
class First2{
First2(){
Frame f=new Frame();

Button b=new Button("click me");
b.setBounds(30,50,80,30);

f.add(b);
f.setSize(300,300);
f.setLayout(null);
f.setVisible(true);
}
public static void main(String args[]){
First2 f=new First2();
}}
```

Event and Listener (Java Event Handling)

Changing the state of an object is known as an event. For example, click on button, dragging mouse etc. The java.awt.event package provides many event classes and Listener interfaces for event handling.

Event classes and Listener interfaces:

Event Classes	Listener Interfaces
ActionEvent	ActionListener
MouseEvent	MouseListener and MouseMotionListener
MouseWheelEvent	MouseWheelListener
KeyEvent	KeyListener
ItemEvent	ItemListener
TextEvent	TextListener
AdjustmentEvent	AdjustmentListener
WindowEvent	WindowListener
ComponentEvent	ComponentListener
ContainerEvent	ContainerListener
FocusEvent	FocusListener

Steps to perform Event Handling

Following steps are required to perform event handling:

1. Implement the Listener interface and overrides its methods
2. Register the component with the Listener

For registering the component with the Listener, many classes provide the registration methods. For example:

- **Button**
  - public void addActionListener(ActionListener a){}
- **MenuItem**
  - public void addActionListener(ActionListener a){}
- **TextField**
  - public void addActionListener(ActionListener a){}
  - public void addTextListener(TextListener a){}
- **TextArea**
  - public void addTextListener(TextListener a){}
- **Checkbox**
  - public void addItemListener(ItemListener a){}
- **Choice**
  - public void addItemListener(ItemListener a){}
- **List**
  - public void addActionListener(ActionListener a){}
  - public void addItemListener(ItemListener a){}

EventHandling Codes:

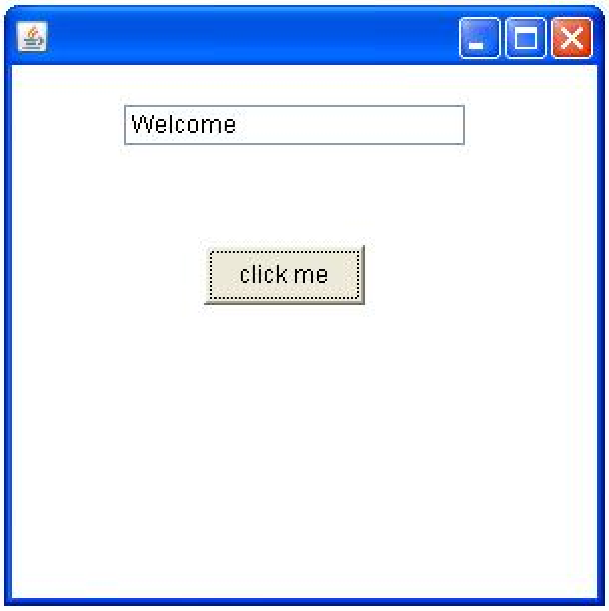
We can put the event handling code into one of the following places:

1. Same class
2. Other class
3. Annonymous class

Example of event handling within class:

```
import java.awt.*;
import java.awt.event.*;

class AEvent extends Frame implements ActionListener{
TextField tf;
AEvent(){

tf=new TextField();
tf.setBounds(60,50,170,20);

Button b=new Button("click me");
b.setBounds(100,120,80,30);

b.addActionListener(this);

add(b);add(tf);

setSize(300,300);
setLayout(null);
setVisible(true);

}

public void actionPerformed(ActionEvent e){
tf.setText("Welcome");
}

public static void main(String args[]){
new AEvent();
}
}
```

public void setBounds(int xaxis, int yaxis, int width, int height); have been used in the above example that sets the position of the component it may be button, textfield etc.



2) Example of event handling by Outer class:

```
import java.awt.*;
import java.awt.event.*;
class AEvent2 extends Frame{
TextField tf;
AEvent2(){

tf=new TextField();
tf.setBounds(60,50,170,20);

Button b=new Button("click me");
b.setBounds(100,120,80,30);

Outer o=new Outer(this);
b.addActionListener(o);//passing outer class instance

add(b);add(tf);

setSize(300,300);
setLayout(null);
setVisible(true);
}
public static void main(String args[]){
new AEvent2();
}
}
```

```
import java.awt.event.*;
class Outer implements ActionListener{
AEvent2 obj;
Outer(AEvent2 obj){
this.obj=obj;
}
public void actionPerformed(ActionEvent e){
obj.tf.setText("welcome");
}
}
```

---

3) Example of event handling by Annonymous class:

```
import java.awt.*;
import java.awt.event.*;
class AEvent3 extends Frame{
TextField tf;
AEvent3(){
tf=new TextField();
tf.setBounds(60,50,170,20);
Button b=new Button("click me");
b.setBounds(50,120,80,30);

b.addActionListener(new ActionListener(){
public void actionPerformed(){
tf.setText("hello");
}
});
add(b);add(tf);
setSize(300,300);
setLayout(null);
setVisible(true);
}
public static void main(String args[]){
new AEvent3();
}
}
```

25. Java Swing Tutorial

26. **Java Swing tutorial** is a part of Java Foundation Classes (JFC) that is *used to create window-based applications*. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.
27. Unlike AWT, Java Swing provides platform-independent and lightweight components.
28. The javax.swing package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.
29.

---

30. Difference between AWT and Swing

31. There are many differences between java awt and swing that are given below.

No.	Java AWT	Java Swing
1)	AWT components are **platform-dependent**.	Java swing components are **platform-**

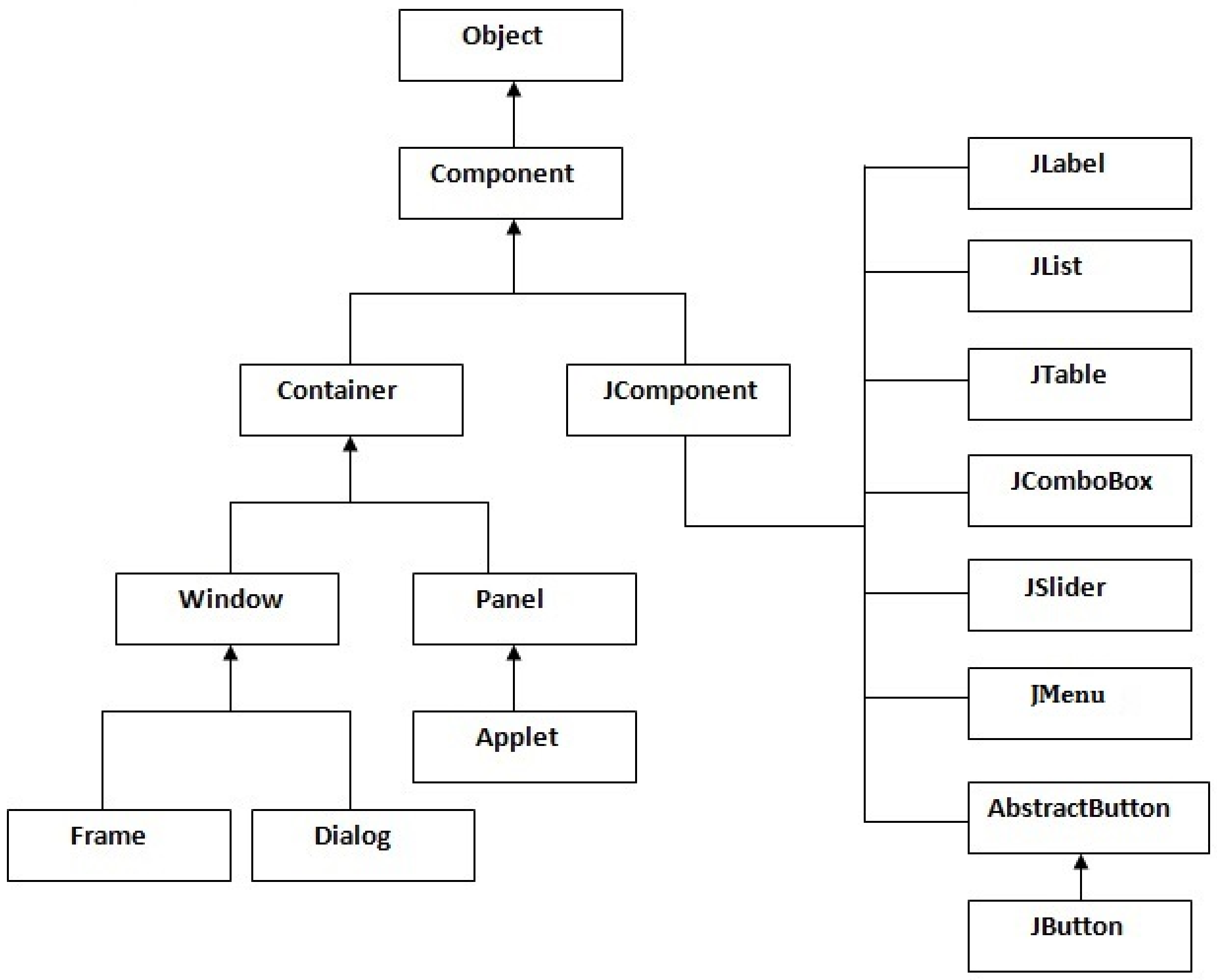
		independent.
2)	AWT components are **heavyweight**.	Swing components are **lightweight**.
3)	AWT **doesn't support pluggable look and feel**.	Swing **supports pluggable look and feel**.
4)	AWT provides **less components** than Swing.	Swing provides **more powerful components**such as tables, lists, scrollpanes, colorchooser, tabbedpane etc.
5)	AWT **doesn't follows MVC**(Model View Controller) where model represents data, view represents presentation and controller acts as an interface between model and view.	Swing **follows MVC**.

32. What is JFC

33. The Java Foundation Classes (JFC) are a set of GUI components which simplify the development of desktop applications.

Hierarchy of Java Swing classes

The hierarchy of java swing API is given below.



Commonly used Methods of Component class

The methods of Component class are widely used in java swing that are given below.

Method	Description
public void add(Component c)	add a component on another component.
public void setSize(int width,int height)	sets size of the component.
public void setLayout(LayoutManager m)	sets the layout manager for the component.
public void setVisible(boolean b)	sets the visibility of the component. It is by default false.

Java Swing Examples

There are two ways to create a frame:

- By creating the object of Frame class (association)
- By extending Frame class (inheritance)

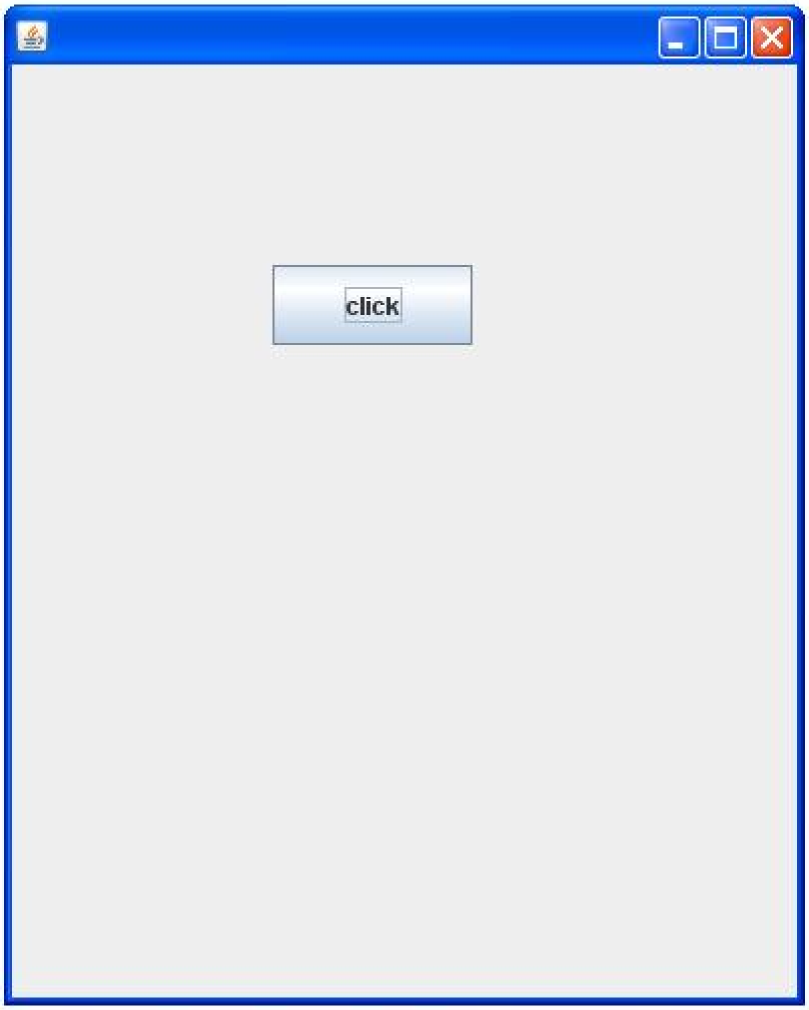
We can write the code of swing inside the main(), constructor or any other method.

Simple Java Swing Example

Let's see a simple swing example where we are creating one button and adding it on the JFrame object inside the main() method.

File: FirstSwingExample.java

```java
import javax.swing.*;
public class FirstSwingExample {
public static void main(String[] args) {
JFrame f=new JFrame();//creating instance of JFrame

JButton b=new JButton("click");//creating instance of JButton
b.setBounds(130,100,100, 40);//x axis, y axis, width, height

f.add(b);//adding button in JFrame

f.setSize(400,500);//400 width and 500 height
f.setLayout(null);//using no layout managers
f.setVisible(true);//making the frame visible
}
}
```



Example of Swing by Association inside constructor

We can also write all the codes of creating JFrame, JButton and method call inside the java constructor.

File: Simple.java

```
import javax.swing.*;
public class Simple {
JFrame f;
Simple(){
f=new JFrame();//creating instance of JFrame

JButton b=new JButton("click");//creating instance of JButton
b.setBounds(130,100,100, 40);

f.add(b);//adding button in JFrame

f.setSize(400,500);//400 width and 500 height
f.setLayout(null);//using no layout managers
f.setVisible(true);//making the frame visible
}

public static void main(String[] args) {
new Simple();
}
}
```

The setBounds(int xaxis, int yaxis, int width, int height)is used in the above example that sets the position of the button.

Simple example of Swing by inheritance

We can also inherit the JFrame class, so there is no need to create the instance of JFrame class explicitly.

File: Simple2.java

```
import javax.swing.*;
public class Simple2 extends JFrame{//inheriting JFrame
JFrame f;
Simple2(){
JButton b=new JButton("click");//create button
b.setBounds(130,100,100, 40);

add(b);//adding button on frame
setSize(400,500);
setLayout(null);
setVisible(true);
}
public static void main(String[] args) {
new Simple2();
}}
```

JButton class

The JButton class is used to create a button that have plateform-independent implementation.

Commonly used Constructors:

- **JButton():** creates a button with no text and icon.
- **JButton(String s):** creates a button with the specified text.
- **JButton(Icon i):** creates a button with the specified icon object.

Commonly used Methods of AbstractButton class:

1) public void setText(String s): is used to set specified text on button.

2) public String getText(): is used to return the text of the button.

3) public void setEnabled(boolean b): is used to enable or disable the button.

4) public void setIcon(Icon b): is used to set the specified Icon on the button.

5) public Icon getIcon(): is used to get the Icon of the button.

6) public void setMnemonic(int a): is used to set the mnemonic on the button.

7) public void addActionListener(ActionListener a): is used to add the action listener to this object.

Note: The JButton class extends AbstractButton class.

Example of displaying image on the button:

```
import java.awt.event.*;
import javax.swing.*;

public class ImageButton{
ImageButton(){
JFrame f=new JFrame();


JButton b=new JButton(new ImageIcon("b.jpg"));
b.setBounds(130,100,100, 40);

f.add(b);

f.setSize(300,400);
f.setLayout(null);
f.setVisible(true);

f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    }

public static void main(String[] args) {
    new ImageButton();
}
}
```

JRadioButton class

The JRadioButton class is used to create a radio button. It is used to choose one option from multiple options. It is widely used in exam systems or quiz.

It should be added in ButtonGroup to select one radio button only.

Commonly used Constructors of JRadioButton class:

- **JRadioButton():** creates an unselected radio button with no text.
- **JRadioButton(String s):** creates an unselected radio button with specified text.
- **JRadioButton(String s, boolean selected):** creates a radio button with the specified text and selected status.

Commonly used Methods of AbstractButton class:

1) public void setText(String s): is used to set specified text on button.

2) public String getText(): is used to return the text of the button.

3) public void setEnabled(boolean b): is used to enable or disable the button.

4) public void setIcon(Icon b): is used to set the specified Icon on the button.

5) public Icon getIcon(): is used to get the Icon of the button.

6) public void setMnemonic(int a): is used to set the mnemonic on the button.

7) public void addActionListener(ActionListener a): is used to add the action listener to this object.

Note: The JRadioButton class extends the JToggleButton class that extends AbstractButton class.

Example of JRadioButton class:

```
import javax.swing.*;
public class Radio {
JFrame f;

Radio(){
f=new JFrame();

JRadioButton r1=new JRadioButton("A) Male");
JRadioButton r2=new JRadioButton("B) FeMale");
r1.setBounds(50,100,70,30);
r2.setBounds(50,150,70,30);

ButtonGroup bg=new ButtonGroup();
bg.add(r1);bg.add(r2);

f.add(r1);f.add(r2);

f.setSize(300,300);
f.setLayout(null);
f.setVisible(true);
}
public static void main(String[] args) {
    new Radio();
}
}
```

download this example

ButtonGroup class:

The ButtonGroup class can be used to group multiple buttons so that at a time only one button can be selected.

JRadioButton example with event handling

```
import javax.swing.*;
import java.awt.event.*;
class RadioExample extends JFrame implements ActionListener{
JRadioButton rb1,rb2;
JButton b;
```

```java
RadioExample(){

rb1=new JRadioButton("Male");
rb1.setBounds(100,50,100,30);

rb2=new JRadioButton("Female");
rb2.setBounds(100,100,100,30);

ButtonGroup bg=new ButtonGroup();
bg.add(rb1);bg.add(rb2);

b=new JButton("click");
b.setBounds(100,150,80,30);
b.addActionListener(this);

add(rb1);add(rb2);add(b);

setSize(300,300);
setLayout(null);
setVisible(true);
}
public void actionPerformed(ActionEvent e){
if(rb1.isSelected()){
JOptionPane.showMessageDialog(this,"You are male");
}
if(rb2.isSelected()){
JOptionPane.showMessageDialog(this,"You are female");
}
}
public static void main(String args[]){
new RadioExample();
}}
```

JTextArea class (Swing Tutorial):

The JTextArea class is used to create a text area. It is a multiline area that displays the plain text only.

Commonly used Constructors:

- **JTextArea():** creates a text area that displays no text initially.
- **JTextArea(String s):** creates a text area that displays specified text initially.
- **JTextArea(int row, int column):** creates a text area with the specified number of rows and columns that displays no text initially..
- **JTextArea(String s, int row, int column):** creates a text area with the specified number of rows and columns that displays specified text.

Commonly used methods of JTextArea class:

1) public void setRows(int rows): is used to set specified number of rows.

2) public void setColumns(int cols):: is used to set specified number of columns.

3) public void setFont(Font f): is used to set the specified font.

4) public void insert(String s, int position): is used to insert the specified text on the specified position.

5) public void append(String s): is used to append the given text to the end of the document.

Example of JTextField class:

```java
import java.awt.Color;
import javax.swing.*;

```

```
public class TArea {
    JTextArea area;
    JFrame f;
    TArea(){
    f=new JFrame();

    area=new JTextArea(300,300);
    area.setBounds(10,30,300,300);

    area.setBackground(Color.black);
    area.setForeground(Color.white);

    f.add(area);

    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
}
    public static void main(String[] args) {
        new TArea();
    }
}
```

JComboBox class:

The JComboBox class is used to create the combobox (drop-down list). At a time only one item can be selected from the item list.

Commonly used Constructors of JComboBox class:

JComboBox()

JComboBox(Object[] items)

JComboBox(Vector<?> items)

Commonly used methods of JComboBox class:

1) public void addItem(Object anObject): is used to add an item to the item list.

2) public void removeItem(Object anObject): is used to delete an item to the item list.

3) public void removeAllItems(): is used to remove all the items from the list.

4) public void setEditable(boolean b): is used to determine whether the JComboBox is editable.

5) public void addActionListener(ActionListener a): is used to add the ActionListener.

6) public void addItemListener(ItemListener i): is used to add the ItemListener.

Example of JComboBox class:

```
import javax.swing.*;
public class Combo {
JFrame f;
Combo(){
    f=new JFrame("Combo ex");

    String country[]={"India","Aus","U.S.A","England","Newzeland"};

    JComboBox cb=new JComboBox(country);
    cb.setBounds(50, 50,90,20);
    f.add(cb);

    f.setLayout(null);
    f.setSize(400,500);
    f.setVisible(true);

}
public static void main(String[] args) {
    new Combo();

}
}
```

JTable class (Swing Tutorial):

The JTable class is used to display the data on two dimensional tables of cells.

Commonly used Constructors of JTable class:

- **JTable():** creates a table with empty cells.
- **JTable(Object[][] rows, Object[] columns):** creates a table with the specified data.

Example of JTable class:

```
import javax.swing.*;
public class MyTable {
    JFrame f;
MyTable(){
    f=new JFrame();

    String data[][]={ {"101","Amit","670000"},
             {"102","Jai","780000"},
                   {"101","Sachin","700000"}};
    String column[]={"ID","NAME","SALARY"};

    JTable jt=new JTable(data,column);
    jt.setBounds(30,40,200,300);

    JScrollPane sp=new JScrollPane(jt);
    f.add(sp);

    f.setSize(300,400);
//  f.setLayout(null);
    f.setVisible(true);
}
public static void main(String[] args) {
    new MyTable();
}
}
```

JColorChooser class:

The JColorChooser class is used to create a color chooser dialog box so that user can select any color.

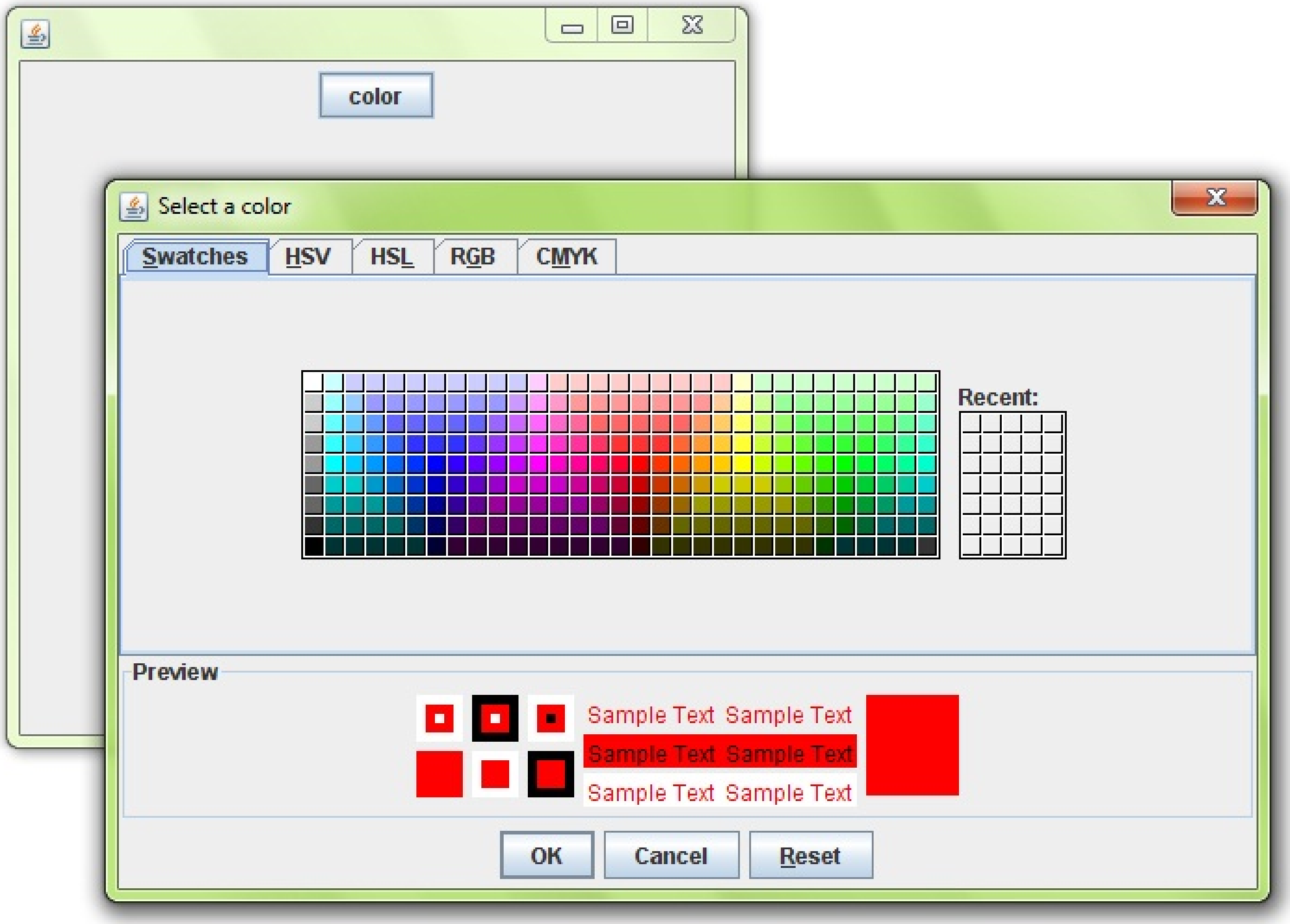
Commonly used Constructors of JColorChooser class:

- **JColorChooser():** is used to create a color chooser pane with white color initially.
- **JColorChooser(Color initialColor):** is used to create a color chooser pane with the specified color initially.

Commonly used methods of JColorChooser class:

public static Color showDialog(Component c, String title, Color initialColor): is used to show the color-chooser dialog box.

Example of JColorChooser class:



```
import java.awt.event.*;
import java.awt.*;
import javax.swing.*;

public class JColorChooserExample extends JFrame implements ActionListener{
JButton b;
Container c;

JColorChooserExample(){
    c=getContentPane();
    c.setLayout(new FlowLayout());

    b=new JButton("color");
    b.addActionListener(this);

```

```
        c.add(b);
    }

    public void actionPerformed(ActionEvent e) {
    Color initialcolor=Color.RED;
    Color color=JColorChooser.showDialog(this,"Select a color",initialcolor);
    c.setBackground(color);
    }

    public static void main(String[] args) {
        JColorChooserExample ch=new JColorChooserExample();
        ch.setSize(400,400);
        ch.setVisible(true);
        ch.setDefaultCloseOperation(EXIT_ON_CLOSE);
    }
    }
```

JProgressBar class:

The JProgressBar class is used to display the progress of the task.

Commonly used Constructors of JProgressBar class:

- **JProgressBar():** is used to create a horizontal progress bar but no string text.
- **JProgressBar(int min, int max):** is used to create a horizontal progress bar with the specified minimum and maximum value.
- **JProgressBar(int orient):** is used to create a progress bar with the specified orientation, it can be either Vertical or Horizontal by using SwingConstants.VERTICAL and SwingConstants.HORIZONTAL constants.
- **JProgressBar(int orient, int min, int max):** is used to create a progress bar with the specified orientation, minimum and maximum value.

Commonly used methods of JProgressBar class:

1) public void setStringPainted(boolean b): is used to determine whether string should be displayed.

2) public void setString(String s): is used to set value to the progress string.

3) public void setOrientation(int orientation): is used to set the orientation, it may be either vertical or horizontal by using SwingConstants.VERTICAL and SwingConstants.HORIZONTAL constants..

4) public void setValue(int value): is used to set the current value on the progress bar.

Example of JProgressBar class:

```
import javax.swing.*;
public class MyProgress extends JFrame{
JProgressBar jb;
int i=0,num=0;

MyProgress(){
jb=new JProgressBar(0,2000);
jb.setBounds(40,40,200,30);

jb.setValue(0);
jb.setStringPainted(true);

add(jb);
setSize(400,400);
setLayout(null);
}

public void iterate(){
```

```
while(i<=2000){
  jb.setValue(i);
  i=i+20;
  try{Thread.sleep(150);}catch(Exception e){}
}
}
public static void main(String[] args) {
    MyProgress m=new MyProgress();
    m.setVisible(true);
    m.iterate();
}
}
```

JSlider class:

The JSlider is used to create the slider. By using JSlider a user can select a value from a specific range.

Commonly used Constructors of JSlider class:

- **JSlider():** creates a slider with the initial value of 50 and range of 0 to 100.
- **JSlider(int orientation):** creates a slider with the specified orientation set by either JSlider.HORIZONTAL or JSlider.VERTICAL with the range 0 to 100 and initial value 50.
- **JSlider(int min, int max):** creates a horizontal slider using the given min and max.
- **JSlider(int min, int max, int value):** creates a horizontal slider using the given min, max and value.
- **JSlider(int orientation, int min, int max, int value):** creates a slider using the given orientation, min, max and value.

Commonly used Methods of JSlider class:

1) public void setMinorTickSpacing(int n): is used to set the minor tick spacing to the slider.

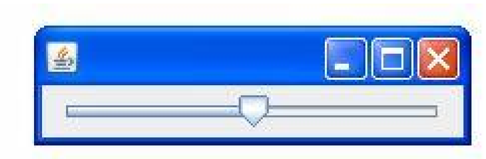
2) public void setMajorTickSpacing(int n): is used to set the major tick spacing to the slider.

3) public void setPaintTicks(boolean b): is used to determine whether tick marks are painted.

4) public void setPaintLabels(boolean b): is used to determine whether labels are painted.

5) public void setPaintTracks(boolean b): is used to determine whether track is painted.

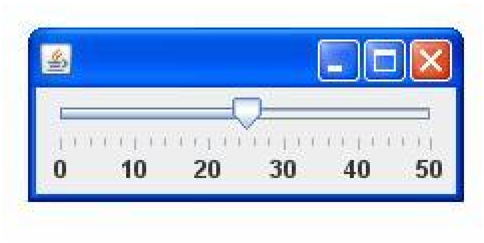
Simple example of JSlider class:



```
import javax.swing.*;

public class SliderExample1 extends JFrame{

public SliderExample1() {
JSlider slider = new JSlider(JSlider.HORIZONTAL, 0, 50, 25);
JPanel panel=new JPanel();
panel.add(slider);

add(panel);
}

public static void main(String s[]) {
SliderExample1 frame=new SliderExample1();
frame.pack();
frame.setVisible(true);
}
}
```

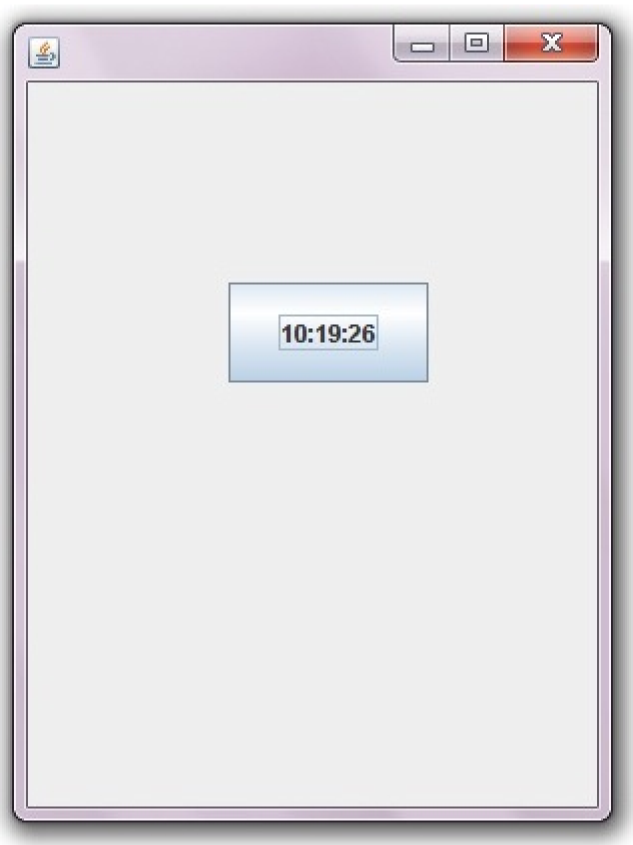
download this example

Example of JSlider class that paints ticks:



```
import javax.swing.*;

public class SliderExample extends JFrame{

public SliderExample() {

JSlider slider = new JSlider(JSlider.HORIZONTAL, 0, 50, 25);
slider.setMinorTickSpacing(2);
slider.setMajorTickSpacing(10);

slider.setPaintTicks(true);
slider.setPaintLabels(true);

JPanel panel=new JPanel();
panel.add(slider);
add(panel);
}

public static void main(String s[]) {
SliderExample frame=new SliderExample();
frame.pack();
frame.setVisible(true);

}
}
```

Example of digital clock in swing:



```
import javax.swing.*;
import java.awt.*;
import java.text.*;
import java.util.*;
public class DigitalWatch implements Runnable{
JFrame f;
Thread t=null;
int hours=0, minutes=0, seconds=0;
String timeString = "";
JButton b;

DigitalWatch(){
   f=new JFrame();

   t = new Thread(this);
      t.start();

   b=new JButton();
      b.setBounds(100,100,100,50);

   f.add(b);
   f.setSize(300,400);
   f.setLayout(null);
   f.setVisible(true);
}

 public void run() {
    try {
      while (true) {

        Calendar cal = Calendar.getInstance();
        hours = cal.get( Calendar.HOUR_OF_DAY );
        if ( hours > 12 ) hours -= 12;
        minutes = cal.get( Calendar.MINUTE );
        seconds = cal.get( Calendar.SECOND );
```

```

                    SimpleDateFormat formatter = new SimpleDateFormat("hh:mm:ss");
                    Date date = cal.getTime();
                    timeString = formatter.format( date );

                    printTime();

                    t.sleep( 1000 );  // interval given in milliseconds
                  }
                }
                catch (Exception e) { }
            }

        public void printTime(){
        b.setText(timeString);
        }

        public static void main(String[] args) {
            new DigitalWatch();


        }
        }
```

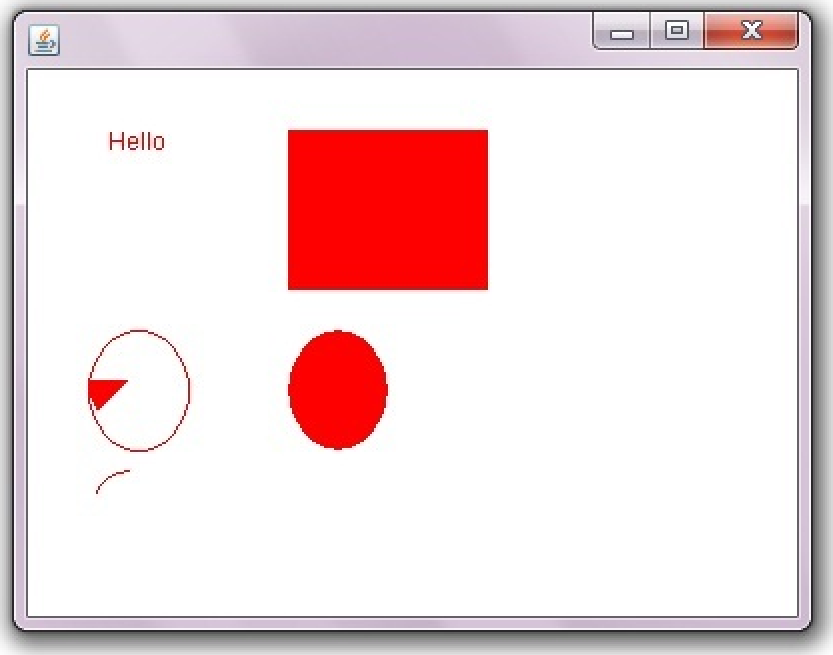
Displaying graphics in swing:

java.awt.Graphics class provides many methods for graphics programming.

Commonly used methods of Graphics class:

1. **public abstract void drawString(String str, int x, int y):** is used to draw the specified string.
2. **public void drawRect(int x, int y, int width, int height):** draws a rectangle with the specified width and height.
3. **public abstract void fillRect(int x, int y, int width, int height):** is used to fill rectangle with the default color and specified width and height.
4. **public abstract void drawOval(int x, int y, int width, int height):** is used to draw oval with the specified width and height.
5. **public abstract void fillOval(int x, int y, int width, int height):** is used to fill oval with the default color and specified width and height.
6. **public abstract void drawLine(int x1, int y1, int x2, int y2):** is used to draw line between the points(x1, y1) and (x2, y2).
7. **public abstract boolean drawImage(Image img, int x, int y, ImageObserver observer):** is used draw the specified image.
8. **public abstract void drawArc(int x, int y, int width, int height, int startAngle, int arcAngle):** is used draw a circular or elliptical arc.
9. **public abstract void fillArc(int x, int y, int width, int height, int startAngle, int arcAngle):** is used to fill a circular or elliptical arc.
10. **public abstract void setColor(Color c):** is used to set the graphics current color to the specified color.
11. **public abstract void setFont(Font font):** is used to set the graphics current font to the specified font.

Example of displaying graphics in swing:



```
import java.awt.*;
import javax.swing.JFrame;

public class DisplayGraphics extends Canvas{

    public void paint(Graphics g) {
        g.drawString("Hello",40,40);
        setBackground(Color.WHITE);
        g.fillRect(130, 30,100, 80);
        g.drawOval(30,130,50, 60);
        setForeground(Color.RED);
        g.fillOval(130,130,50, 60);
        g.drawArc(30, 200, 40,50,90,60);
        g.fillArc(30, 130, 40,50,180,40);

    }
        public static void main(String[] args) {
        DisplayGraphics m=new DisplayGraphics();
        JFrame f=new JFrame();
        f.add(m);
        f.setSize(400,400);
        //f.setLayout(null);
        f.setVisible(true);
    }

}
```

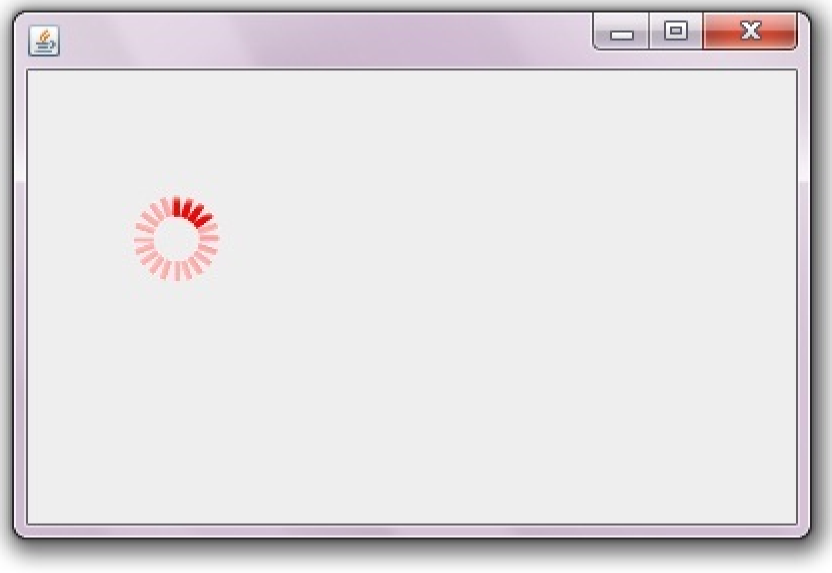
Displaying image in swing:

For displaying image, we can use the method drawImage() of Graphics class.

Syntax of drawImage() method:

1. **public abstract boolean drawImage(Image img, int x, int y, ImageObserver observer):** is used draw the specified image.

Example of displaying image in swing:



```
import java.awt.*;
import javax.swing.JFrame;

public class MyCanvas extends Canvas{

    public void paint(Graphics g) {

        Toolkit t=Toolkit.getDefaultToolkit();
        Image i=t.getImage("p3.gif");
        g.drawImage(i, 120,100,this);

    }
        public static void main(String[] args) {
        MyCanvas m=new MyCanvas();
        JFrame f=new JFrame();
        f.add(m);
        f.setSize(400,400);
        f.setVisible(true);
    }

}
```

Example of creating Edit menu for Notepad:

```
import javax.swing.*;
import java.awt.event.*;

public class Notepad implements ActionListener{
JFrame f;
JMenuBar mb;
JMenu file,edit,help;
JMenuItem cut,copy,paste,selectAll;
JTextArea ta;

Notepad(){
f=new JFrame();

cut=new JMenuItem("cut");
copy=new JMenuItem("copy");
paste=new JMenuItem("paste");
selectAll=new JMenuItem("selectAll");

cut.addActionListener(this);
copy.addActionListener(this);
paste.addActionListener(this);
selectAll.addActionListener(this);

mb=new JMenuBar();
mb.setBounds(5,5,400,40);

file=new JMenu("File");
edit=new JMenu("Edit");
help=new JMenu("Help");

edit.add(cut);edit.add(copy);edit.add(paste);edit.add(selectAll);


mb.add(file);mb.add(edit);mb.add(help);

ta=new JTextArea();
ta.setBounds(5,30,460,460);

f.add(mb);f.add(ta);

f.setLayout(null);
f.setSize(500,500);
f.setVisible(true);
}

public void actionPerformed(ActionEvent e) {
if(e.getSource()==cut)
ta.cut();
if(e.getSource()==paste)
ta.paste();
if(e.getSource()==copy)
ta.copy();
if(e.getSource()==selectAll)
ta.selectAll();
}

public static void main(String[] args) {
    new Notepad();
}
}
```

Example of open dialog box:

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.io.*;

public class OpenMenu extends JFrame implements ActionListener{
JMenuBar mb;
JMenu file;
JMenuItem open;
JTextArea ta;
OpenMenu(){
open=new JMenuItem("Open File");
open.addActionListener(this);

file=new JMenu("File");
file.add(open);

mb=new JMenuBar();
mb.setBounds(0,0,800,20);
mb.add(file);

ta=new JTextArea(800,800);
ta.setBounds(0,20,800,800);

add(mb);
add(ta);

}
public void actionPerformed(ActionEvent e) {
if(e.getSource()==open){
openFile();
}
}

void openFile(){
JFileChooser fc=new JFileChooser();
int i=fc.showOpenDialog(this);

if(i==JFileChooser.APPROVE_OPTION){
File f=fc.getSelectedFile();
String filepath=f.getPath();

displayContent(filepath);

}

}

void displayContent(String fpath){
try{
BufferedReader br=new BufferedReader(new FileReader(fpath));
String s1="",s2="";

while((s1=br.readLine())!=null){
s2+=s1+"\n";
}
ta.setText(s2);
br.close();
}catch (Exception e) {e.printStackTrace();  }
}

public static void main(String[] args) {
    OpenMenu om=new OpenMenu();
    om.setSize(800,800);
    om.setLayout(null);
    om.setVisible(true);
    om.setDefaultCloseOperation(EXIT_ON_CLOSE);
}
}
```

Notepad in Java with source code

Notepad in Java with source code: We can develop Notepad in java with the help of AWT/Swing with event handling. Let's see the code of creating Notepad in java.

```
import java.io.*;
import java.util.Date;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;

/************************************/
class FileOperation
{
Notepad npd;

boolean saved;
boolean newFileFlag;
String fileName;
String applicationTitle="Notepad - JavaTpoint";

File fileRef;
JFileChooser chooser;
/////////////////////////////
boolean isSave(){return saved;}
void setSave(boolean saved){this.saved=saved;}
String getFileName(){return new String(fileName);}
void setFileName(String fileName){this.fileName=new String(fileName);}
/////////////////////////
FileOperation(Notepad npd)
{
this.npd=npd;

saved=true;
newFileFlag=true;
fileName=new String("Untitled");
fileRef=new File(fileName);
this.npd.f.setTitle(fileName+" - "+applicationTitle);

chooser=new JFileChooser();
chooser.addChoosableFileFilter(new MyFileFilter(".java","Java Source Files(*.java)"));
chooser.addChoosableFileFilter(new MyFileFilter(".txt","Text Files(*.txt)"));
chooser.setCurrentDirectory(new File("."));

}
//////////////////////////////////////

boolean saveFile(File temp)
{
FileWriter fout=null;
try
{
fout=new FileWriter(temp);
fout.write(npd.ta.getText());
}
catch(IOException ioe){updateStatus(temp,false);return false;}
finally
{try{fout.close();}catch(IOException excp){}}
updateStatus(temp,true);
return true;
}
///////////////////////
boolean saveThisFile()
{

if(!newFileFlag)
    {return saveFile(fileRef);}

return saveAsFile();
}
////////////////////////////////////
boolean saveAsFile()
{
File temp=null;
chooser.setDialogTitle("Save As...");
chooser.setApproveButtonText("Save Now");
chooser.setApproveButtonMnemonic(KeyEvent.VK_S);
chooser.setApproveButtonToolTipText("Click me to save!");

do
{
if(chooser.showSaveDialog(this.npd.f)!=JFileChooser.APPROVE_OPTION)
    return false;
temp=chooser.getSelectedFile();
if(!temp.exists()) break;
if(   JOptionPane.showConfirmDialog(
    this.npd.f,"<html>"+temp.getPath()+" already exists.<br>Do you want to replace it?<html>",
    "Save As",JOptionPane.YES_NO_OPTION
```

```
                        )==JOptionPane.YES_OPTION)
            break;
        }while(true);


        return saveFile(temp);
        }

        ///////////////////////
        boolean openFile(File temp)
        {
        FileInputStream fin=null;
        BufferedReader din=null;

        try
        {
        fin=new FileInputStream(temp);
        din=new BufferedReader(new InputStreamReader(fin));
        String str=" ";
        while(str!=null)
        {
        str=din.readLine();
        if(str==null)
        break;
        this.npd.ta.append(str+"\n");
        }

        }
        catch(IOException ioe){updateStatus(temp,false);return false;}
        finally
        {try{din.close();fin.close();}catch(IOException excp){}}
        updateStatus(temp,true);
        this.npd.ta.setCaretPosition(0);
        return true;
        }
        ///////////////////////
        void openFile()
        {
        if(!confirmSave()) return;
        chooser.setDialogTitle("Open File...");
        chooser.setApproveButtonText("Open this");
        chooser.setApproveButtonMnemonic(KeyEvent.VK_O);
        chooser.setApproveButtonToolTipText("Click me to open the selected file.!");

        File temp=null;
        do
        {
        if(chooser.showOpenDialog(this.npd.f)!=JFileChooser.APPROVE_OPTION)
            return;
        temp=chooser.getSelectedFile();

        if(temp.exists())   break;

        JOptionPane.showMessageDialog(this.npd.f,
            "<html>"+temp.getName()+"<br>file not found.<br>"+
            "Please verify the correct file name was given.<html>",
            "Open", JOptionPane.INFORMATION_MESSAGE);

        } while(true);

        this.npd.ta.setText("");

        if(!openFile(temp))
            {
            fileName="Untitled"; saved=true;
            this.npd.f.setTitle(fileName+" - "+applicationTitle);
            }
        if(!temp.canWrite())
            newFileFlag=true;

        }
```

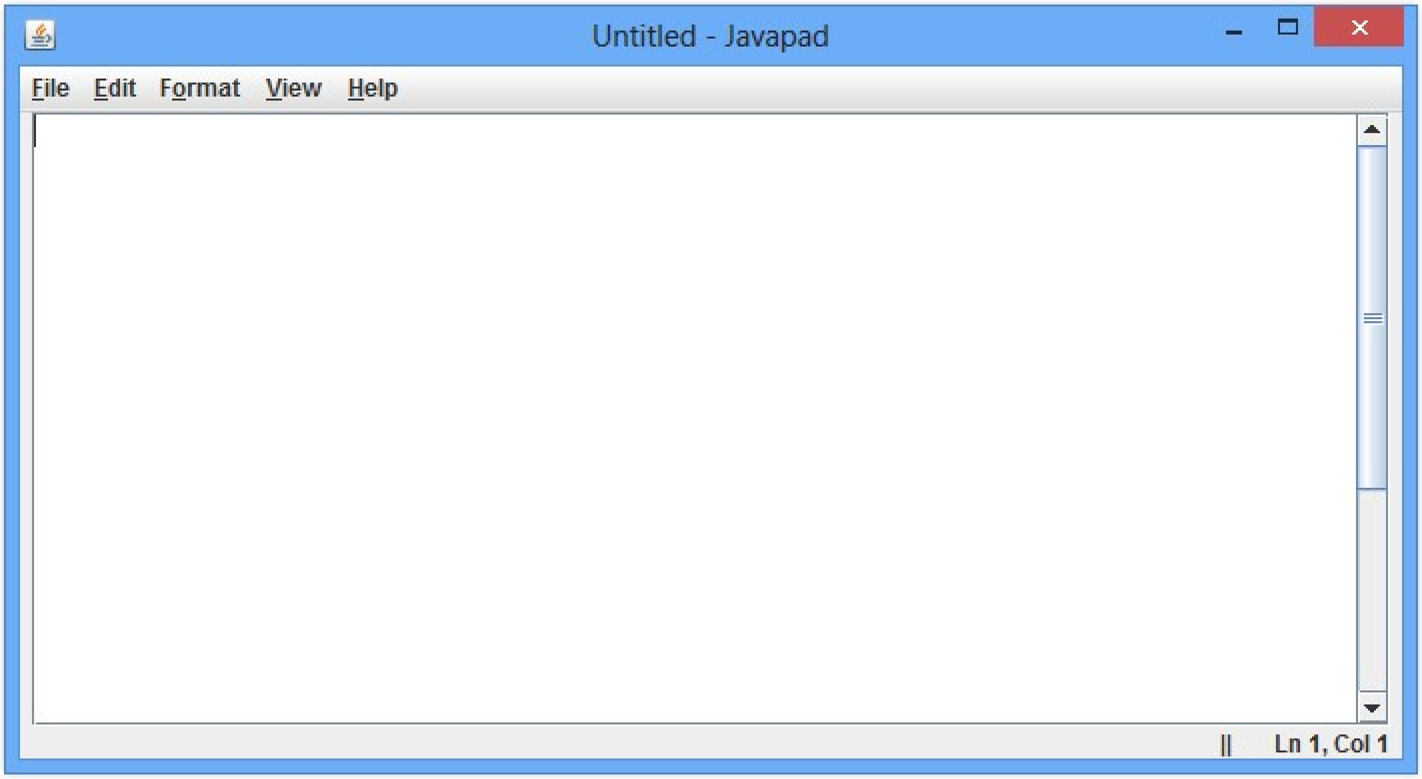
```java
//////////////////////
void updateStatus(File temp,boolean saved)
{
if(saved)
{
this.saved=true;
fileName=new String(temp.getName());
if(!temp.canWrite())
    {fileName+="(Read only)"; newFileFlag=true;}
fileRef=temp;
npd.f.setTitle(fileName + " - "+applicationTitle);
npd.statusBar.setText("File : "+temp.getPath()+" saved/opened successfully.");
newFileFlag=false;
}
else
{
npd.statusBar.setText("Failed to save/open : "+temp.getPath());
}
}
/////////////////////
boolean confirmSave()
{
String strMsg="<html>The text in the "+fileName+" file has been changed.<br>"+
    "Do you want to save the changes?<html>";
if(!saved)
{
int x=JOptionPane.showConfirmDialog(this.npd.f,strMsg,applicationTitle,
JOptionPane.YES_NO_CANCEL_OPTION);

if(x==JOptionPane.CANCEL_OPTION) return false;
if(x==JOptionPane.YES_OPTION && !saveAsFile()) return false;
}
return true;
}
///////////////////////////////////
void newFile()
{
if(!confirmSave()) return;

this.npd.ta.setText("");
fileName=new String("Untitled");
fileRef=new File(fileName);
saved=true;
newFileFlag=true;
this.npd.f.setTitle(fileName+" - "+applicationTitle);
}
//////////////////////////////////
}// end defination of class FileOperation
/************************************/
public class Notepad  implements ActionListener, MenuConstants
{

JFrame f;
JTextArea ta;
JLabel statusBar;

private String fileName="Untitled";
private boolean saved=true;
String applicationName="Javapad";

String searchString, replaceString;
int lastSearchIndex;

FileOperation fileHandler;
FontChooser fontDialog=null;
FindDialog findReplaceDialog=null;
JColorChooser bcolorChooser=null;
JColorChooser fcolorChooser=null;
JDialog backgroundDialog=null;
JDialog foregroundDialog=null;
JMenuItem cutItem,copyItem, deleteItem, findItem, findNextItem,
replaceItem, gotoItem, selectAllItem;
/****************************/
Notepad()
{
f=new JFrame(fileName+" - "+applicationName);
ta=new JTextArea(30,60);
statusBar=new JLabel("||       Ln 1, Col 1  ",JLabel.RIGHT);
f.add(new JScrollPane(ta),BorderLayout.CENTER);
f.add(statusBar,BorderLayout.SOUTH);
f.add(new JLabel("  "),BorderLayout.EAST);
f.add(new JLabel("  "),BorderLayout.WEST);
createMenuBar(f);
//f.setSize(350,350);
f.pack();
f.setLocation(100,50);
f.setVisible(true);
f.setLocation(150,50);
f.setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE);

fileHandler=new FileOperation(this);

////////////////////

ta.addCaretListener(
new CaretListener()
{
public void caretUpdate(CaretEvent e)
{
int lineNumber=0, column=0, pos=0;

try
{
pos=ta.getCaretPosition();
lineNumber=ta.getLineOfOffset(pos);
column=pos-ta.getLineStartOffset(lineNumber);
}catch(Exception excp){}
if(ta.getText().length()==0){lineNumber=0; column=0;}
statusBar.setText("||       Ln "+(lineNumber+1)+", Col "+(column+1));
}
});
/////////////////
DocumentListener myListener = new DocumentListener()
{
public void changedUpdate(DocumentEvent e){fileHandler.saved=false;}
public void removeUpdate(DocumentEvent e){fileHandler.saved=false;}
public void insertUpdate(DocumentEvent e){fileHandler.saved=false;}
};
ta.getDocument().addDocumentListener(myListener);
/////////
WindowListener frameClose=new WindowAdapter()
{
public void windowClosing(WindowEvent we)
{
if(fileHandler.confirmSave())System.exit(0);
}
};
f.addWindowListener(frameClose);
/////////////////
/*
ta.append("Hello dear hello hi");
ta.append("\nwho are u dear mister hello");
ta.append("\nhello bye hel");
ta.append("\nHello");
ta.append("\nMiss u mister hello hell");
fileHandler.saved=true;
*/
}
///////////////////////////////////
void goTo()
{
int lineNumber=0;
```

```
try
{
lineNumber=ta.getLineOfOffset(ta.getCaretPosition())+1;
String tempStr=JOptionPane.showInputDialog(f,"Enter Line Number:",""+lineNumber);
if(tempStr==null)
    {return;}
lineNumber=Integer.parseInt(tempStr);
ta.setCaretPosition(ta.getLineStartOffset(lineNumber-1));
}catch(Exception e){}
}
//////////////////////////////////
public void actionPerformed(ActionEvent ev)
{
String cmdText=ev.getActionCommand();
////////////////////////////////////
if(cmdText.equals(fileNew))
    fileHandler.newFile();
else if(cmdText.equals(fileOpen))
    fileHandler.openFile();
////////////////////////////////////
else if(cmdText.equals(fileSave))
    fileHandler.saveThisFile();
////////////////////////////////////
else if(cmdText.equals(fileSaveAs))
    fileHandler.saveAsFile();
////////////////////////////////////
else if(cmdText.equals(fileExit))
    {if(fileHandler.confirmSave())System.exit(0);}
////////////////////////////////////
else if(cmdText.equals(filePrint))
JOptionPane.showMessageDialog(
    Notepad.this.f,
    "Get ur printer repaired first! It seems u dont have one!",
    "Bad Printer",
    JOptionPane.INFORMATION_MESSAGE
    );
////////////////////////////////////
else if(cmdText.equals(editCut))
    ta.cut();
////////////////////////////////////
else if(cmdText.equals(editCopy))
    ta.copy();
////////////////////////////////////
else if(cmdText.equals(editPaste))
    ta.paste();
////////////////////////////////////
else if(cmdText.equals(editDelete))
    ta.replaceSelection("");
////////////////////////////////////
else if(cmdText.equals(editFind))
{
if(Notepad.this.ta.getText().length()==0)
    return; // text box have no text
if(findReplaceDialog==null)
    findReplaceDialog=new FindDialog(Notepad.this.ta);
findReplaceDialog.showDialog(Notepad.this.f,true);//find
}
////////////////////////////////////
else if(cmdText.equals(editFindNext))
{
if(Notepad.this.ta.getText().length()==0)
    return; // text box have no text

if(findReplaceDialog==null)
    statusBar.setText("Use Find option of Edit Menu first !!!!");
else
    findReplaceDialog.findNextWithSelection();
}
////////////////////////////////////
else if(cmdText.equals(editReplace))
{
if(Notepad.this.ta.getText().length()==0)
    return; // text box have no text

if(findReplaceDialog==null)
    findReplaceDialog=new FindDialog(Notepad.this.ta);
findReplaceDialog.showDialog(Notepad.this.f,false);//replace
}
////////////////////////////////////
else if(cmdText.equals(editGoTo))
{
if(Notepad.this.ta.getText().length()==0)
    return; // text box have no text
goTo();
}
////////////////////////////////////
else if(cmdText.equals(editSelectAll))
    ta.selectAll();
////////////////////////////////////
else if(cmdText.equals(editTimeDate))
    ta.insert(new Date().toString(),ta.getSelectionStart());
////////////////////////////////////
else if(cmdText.equals(formatWordWrap))
{
JCheckBoxMenuItem temp=(JCheckBoxMenuItem)ev.getSource();
ta.setLineWrap(temp.isSelected());
}
////////////////////////////////////
else if(cmdText.equals(formatFont))
{
if(fontDialog==null)
    fontDialog=new FontChooser(ta.getFont());

if(fontDialog.showDialog(Notepad.this.f,"Choose a font"))
    Notepad.this.ta.setFont(fontDialog.createFont());
}
////////////////////////////////////
else if(cmdText.equals(formatForeground))
    showForegroundColorDialog();
////////////////////////////////////
else if(cmdText.equals(formatBackground))
    showBackgroundColorDialog();
////////////////////////////////////

else if(cmdText.equals(viewStatusBar))
{
JCheckBoxMenuItem temp=(JCheckBoxMenuItem)ev.getSource();
statusBar.setVisible(temp.isSelected());
}
////////////////////////////////////
else if(cmdText.equals(helpAboutNotepad))
{
JOptionPane.showMessageDialog(Notepad.this.f,aboutText,"Dedicated 2 u!",
JOptionPane.INFORMATION_MESSAGE);
}
else
    statusBar.setText("This "+cmdText+" command is yet to be implemented");
}//action Performed
////////////////////////////////////
void showBackgroundColorDialog()
{
if(bcolorChooser==null)
    bcolorChooser=new JColorChooser();
if(backgroundDialog==null)
    backgroundDialog=JColorChooser.createDialog
        (Notepad.this.f,
        formatBackground,
        false,
        bcolorChooser,
        new ActionListener()
        {public void actionPerformed(ActionEvent evvv){
            Notepad.this.ta.setBackground(bcolorChooser.getColor());}},
        null);

backgroundDialog.setVisible(true);
}
///////////////////////////////////
void showForegroundColorDialog()
{
if(fcolorChooser==null)
    fcolorChooser=new JColorChooser();
if(foregroundDialog==null)
    foregroundDialog=JColorChooser.createDialog
        (Notepad.this.f,
        formatForeground,
        false,
        fcolorChooser,
        new ActionListener()
        {public void actionPerformed(ActionEvent evvv){
            Notepad.this.ta.setForeground(fcolorChooser.getColor());}},
        null);

foregroundDialog.setVisible(true);
}

///////////////////////////////////
JMenuItem createMenuItem(String s, int key,JMenu toMenu,ActionListener al)
{
JMenuItem temp=new JMenuItem(s,key);
temp.addActionListener(al);
toMenu.add(temp);

return temp;
}
///////////////////////////////////
JMenuItem createMenuItem(String s, int key,JMenu toMenu,int aclKey,ActionListener al)
{
JMenuItem temp=new JMenuItem(s,key);
temp.addActionListener(al);
temp.setAccelerator(KeyStroke.getKeyStroke(aclKey,ActionEvent.CTRL_MASK));
toMenu.add(temp);

return temp;
}
///////////////////////////////////
JCheckBoxMenuItem createCheckBoxMenuItem(String s,
 int key,JMenu toMenu,ActionListener al)
{
JCheckBoxMenuItem temp=new JCheckBoxMenuItem(s);
temp.setMnemonic(key);
temp.addActionListener(al);
temp.setSelected(false);
toMenu.add(temp);

return temp;
}
///////////////////////////////////
JMenu createMenu(String s,int key,JMenuBar toMenuBar)
{
JMenu temp=new JMenu(s);
temp.setMnemonic(key);
toMenuBar.add(temp);
return temp;
}
/*********************************/
void createMenuBar(JFrame f)
{
JMenuBar mb=new JMenuBar();
JMenuItem temp;

JMenu fileMenu=createMenu(fileText,KeyEvent.VK_F,mb);
JMenu editMenu=createMenu(editText,KeyEvent.VK_E,mb);
JMenu formatMenu=createMenu(formatText,KeyEvent.VK_O,mb);
```

```
JMenu viewMenu=createMenu(viewText,KeyEvent.VK_V,mb);
JMenu helpMenu=createMenu(helpText,KeyEvent.VK_H,mb);

createMenuItem(fileNew,KeyEvent.VK_N,fileMenu,KeyEvent.VK_N,this);
createMenuItem(fileOpen,KeyEvent.VK_O,fileMenu,KeyEvent.VK_O,this);
createMenuItem(fileSave,KeyEvent.VK_S,fileMenu,KeyEvent.VK_S,this);
createMenuItem(fileSaveAs,KeyEvent.VK_A,fileMenu,this);
fileMenu.addSeparator();
temp=createMenuItem(filePageSetup,KeyEvent.VK_U,fileMenu,this);
temp.setEnabled(false);
createMenuItem(filePrint,KeyEvent.VK_P,fileMenu,KeyEvent.VK_P,this);
fileMenu.addSeparator();
createMenuItem(fileExit,KeyEvent.VK_X,fileMenu,this);

temp=createMenuItem(editUndo,KeyEvent.VK_U,editMenu,KeyEvent.VK_Z,this);
temp.setEnabled(false);
editMenu.addSeparator();
cutItem=createMenuItem(editCut,KeyEvent.VK_T,editMenu,KeyEvent.VK_X,this);
copyItem=createMenuItem(editCopy,KeyEvent.VK_C,editMenu,KeyEvent.VK_C,this);
createMenuItem(editPaste,KeyEvent.VK_P,editMenu,KeyEvent.VK_V,this);
deleteItem=createMenuItem(editDelete,KeyEvent.VK_L,editMenu,this);
deleteItem.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_DELETE,0));
editMenu.addSeparator();
findItem=createMenuItem(editFind,KeyEvent.VK_F,editMenu,KeyEvent.VK_F,this);
findNextItem=createMenuItem(editFindNext,KeyEvent.VK_N,editMenu,this);
findNextItem.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_F3,0));
replaceItem=createMenuItem(editReplace,KeyEvent.VK_R,editMenu,KeyEvent.VK_H,this);
gotoItem=createMenuItem(editGoTo,KeyEvent.VK_G,editMenu,KeyEvent.VK_G,this);
editMenu.addSeparator();
selectAllItem=createMenuItem(editSelectAll,KeyEvent.VK_A,editMenu,KeyEvent.VK_A,this);
createMenuItem(editTimeDate,KeyEvent.VK_D,editMenu,this)
.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_F5,0));

createCheckBoxMenuItem(formatWordWrap,KeyEvent.VK_W,formatMenu,this);

createMenuItem(formatFont,KeyEvent.VK_F,formatMenu,this);
formatMenu.addSeparator();
createMenuItem(formatForeground,KeyEvent.VK_T,formatMenu,this);
createMenuItem(formatBackground,KeyEvent.VK_P,formatMenu,this);

createCheckBoxMenuItem(viewStatusBar,KeyEvent.VK_S,viewMenu,this).setSelected(true);
/************For Look and Feel***/
LookAndFeelMenu.createLookAndFeelMenuItem(viewMenu,this.f);


temp=createMenuItem(helpHelpTopic,KeyEvent.VK_H,helpMenu,this);
temp.setEnabled(false);
helpMenu.addSeparator();
createMenuItem(helpAboutNotepad,KeyEvent.VK_A,helpMenu,this);

MenuListener editMenuListener=new MenuListener()
{
   public void menuSelected(MenuEvent evvvv)
    {
    if(Notepad.this.ta.getText().length()==0)
    {
    findItem.setEnabled(false);
    findNextItem.setEnabled(false);
    replaceItem.setEnabled(false);
    selectAllItem.setEnabled(false);
    gotoItem.setEnabled(false);
    }
    else
    {
    findItem.setEnabled(true);
    findNextItem.setEnabled(true);
    replaceItem.setEnabled(true);
    selectAllItem.setEnabled(true);
    gotoItem.setEnabled(true);
    }
    if(Notepad.this.ta.getSelectionStart()==ta.getSelectionEnd())
```

```
        {
        cutItem.setEnabled(false);
        copyItem.setEnabled(false);
        deleteItem.setEnabled(false);
        }
        else
        {
        cutItem.setEnabled(true);
        copyItem.setEnabled(true);
        deleteItem.setEnabled(true);
        }
        }
       public void menuDeselected(MenuEvent evvvv){}
       public void menuCanceled(MenuEvent evvvv){}
    };
    editMenu.addMenuListener(editMenuListener);
    f.setJMenuBar(mb);
    }
    /*************Constructor**************/
    ////////////////////////////////////
    public static void main(String[] s)
    {
    new Notepad();
    }
    }
    /**************************************/
    //public
    interface MenuConstants
    {
    final String fileText="File";
    final String editText="Edit";
    final String formatText="Format";
    final String viewText="View";
    final String helpText="Help";

    final String fileNew="New";
    final String fileOpen="Open...";
    final String fileSave="Save";
    final String fileSaveAs="Save As...";
    final String filePageSetup="Page Setup...";
    final String filePrint="Print";
    final String fileExit="Exit";

    final String editUndo="Undo";
    final String editCut="Cut";
    final String editCopy="Copy";
    final String editPaste="Paste";
    final String editDelete="Delete";
    final String editFind="Find...";
    final String editFindNext="Find Next";
    final String editReplace="Replace";
    final String editGoTo="Go To...";
    final String editSelectAll="Select All";
    final String editTimeDate="Time/Date";

    final String formatWordWrap="Word Wrap";
    final String formatFont="Font...";
    final String formatForeground="Set Text color...";
    final String formatBackground="Set Pad color...";

    final String viewStatusBar="Status Bar";

    final String helpHelpTopic="Help Topic";
    final String helpAboutNotepad="About Javapad";

    final String aboutText="Your Javapad";
    }
```

download this example



Calculator in Java with Source Code

Calculator in Java with Source Code: We can develop calculator in java with the help of AWT/Swing with event handling. Let's see the code of creating calculator in java.

```
/*********************************************
Save this file as MyCalculator.java
to compile it use
    javac MyCalculator.java
to use the calcuator do this
    java MyCalculator

**********************************************/
import java.awt.*;
import java.awt.event.*;
/*********************************************/

public class MyCalculator extends Frame
{

public boolean setClear=true;
double number, memValue;
char op;

String digitButtonText[] = {"7", "8", "9", "4", "5", "6", "1", "2", "3", "0", "+/-", "." };
String operatorButtonText[] = {"/", "sqrt", "*", "%", "-", "1/X", "+", "=" };
String memoryButtonText[] = {"MC", "MR", "MS", "M+" };
String specialButtonText[] = {"Backspc", "C", "CE" };

MyDigitButton digitButton[]=new MyDigitButton[digitButtonText.length];
MyOperatorButton operatorButton[]=new MyOperatorButton[operatorButtonText.length];
MyMemoryButton memoryButton[]=new MyMemoryButton[memoryButtonText.length];
MySpecialButton specialButton[]=new MySpecialButton[specialButtonText.length];

```

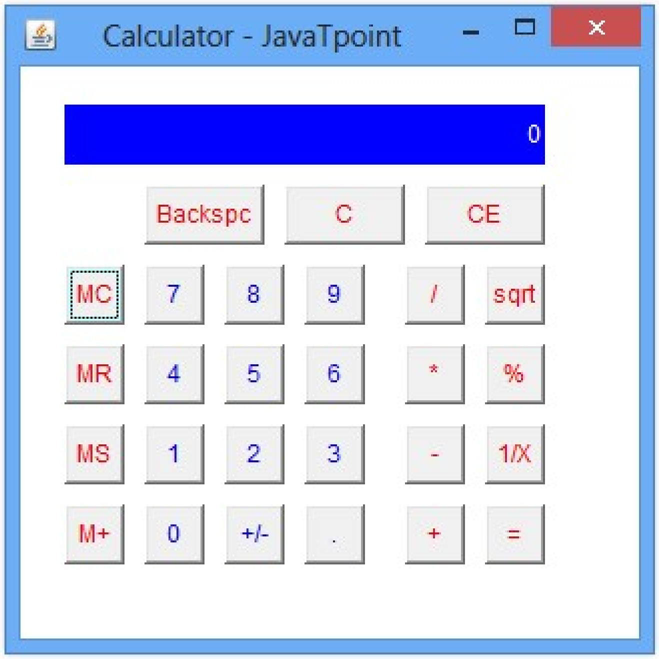
```java
Label displayLabel=new Label("0",Label.RIGHT);
Label memLabel=new Label(" ",Label.RIGHT);

final int FRAME_WIDTH=325,FRAME_HEIGHT=325;
final int HEIGHT=30, WIDTH=30, H_SPACE=10,V_SPACE=10;
final int TOPX=30, TOPY=50;
///////////////////////////
MyCalculator(String frameText)//constructor
{
super(frameText);

int tempX=TOPX, y=TOPY;
displayLabel.setBounds(tempX,y,240,HEIGHT);
displayLabel.setBackground(Color.BLUE);
displayLabel.setForeground(Color.WHITE);
add(displayLabel);

memLabel.setBounds(TOPX,  TOPY+HEIGHT+ V_SPACE,WIDTH, HEIGHT);
add(memLabel);

// set Co-ordinates for Memory Buttons
tempX=TOPX;
y=TOPY+2*(HEIGHT+V_SPACE);
for(int i=0; i<memoryButton.length; i++)
{
memoryButton[i]=new MyMemoryButton(tempX,y,WIDTH,HEIGHT,memoryButtonText[i], this);
memoryButton[i].setForeground(Color.RED);
y+=HEIGHT+V_SPACE;
}

//set Co-ordinates for Special Buttons
tempX=TOPX+1*(WIDTH+H_SPACE); y=TOPY+1*(HEIGHT+V_SPACE);
for(int i=0;i<specialButton.length;i++)
{
specialButton[i]=new MySpecialButton(tempX,y,WIDTH*2,HEIGHT,specialButtonText[i], this);
specialButton[i].setForeground(Color.RED);
tempX=tempX+2*WIDTH+H_SPACE;
}

//set Co-ordinates for Digit Buttons
int digitX=TOPX+WIDTH+H_SPACE;
int digitY=TOPY+2*(HEIGHT+V_SPACE);
tempX=digitX;  y=digitY;
for(int i=0;i<digitButton.length;i++)
{
digitButton[i]=new MyDigitButton(tempX,y,WIDTH,HEIGHT,digitButtonText[i], this);
digitButton[i].setForeground(Color.BLUE);
tempX+=WIDTH+H_SPACE;
if((i+1)%3==0){tempX=digitX; y+=HEIGHT+V_SPACE;}
}

//set Co-ordinates for Operator Buttons
int opsX=digitX+2*(WIDTH+H_SPACE)+H_SPACE;
int opsY=digitY;
tempX=opsX;  y=opsY;
for(int i=0;i<operatorButton.length;i++)
{
tempX+=WIDTH+H_SPACE;
operatorButton[i]=new MyOperatorButton(tempX,y,WIDTH,HEIGHT,operatorButtonText[i], this);
operatorButton[i].setForeground(Color.RED);
if((i+1)%2==0){tempX=opsX; y+=HEIGHT+V_SPACE;}
}

addWindowListener(new WindowAdapter()
{
public void windowClosing(WindowEvent ev)
{System.exit(0);}
});

setLayout(null);
setSize(FRAME_WIDTH,FRAME_HEIGHT);
setVisible(true);
}
//////////////////////////////////
static String getFormattedText(double temp)
{
String resText=""+temp;
if(resText.lastIndexOf(".0")>0)
    resText=resText.substring(0,resText.length()-2);
return resText;
}
////////////////////////////////////////
public static void main(String []args)
{
new MyCalculator("Calculator - JavaTpoint");
}
}

/********************************************/

class MyDigitButton extends Button implements ActionListener
{
MyCalculator cl;

//////////////////////////////////////////
MyDigitButton(int x,int y, int width,int height,String cap, MyCalculator clc)
{
super(cap);
setBounds(x,y,width,height);
this.cl=clc;
this.cl.add(this);
addActionListener(this);
}
////////////////////////////////////////////////
static boolean isInString(String s, char ch)
{
for(int i=0; i<s.length();i++) if(s.charAt(i)==ch) return true;
return false;
}
/////////////////////////////////////////////////
public void actionPerformed(ActionEvent ev)
{
String tempText=((MyDigitButton)ev.getSource()).getLabel();

if(tempText.equals("."))
{
 if(cl.setClear)
    {cl.displayLabel.setText("0.");cl.setClear=false;}
 else if(!isInString(cl.displayLabel.getText(),'.'))
    cl.displayLabel.setText(cl.displayLabel.getText()+".");
 return;
}

int index=0;
try{
        index=Integer.parseInt(tempText);
    }catch(NumberFormatException e){return;}

if (index==0 && cl.displayLabel.getText().equals("0")) return;

if(cl.setClear)
            {cl.displayLabel.setText(""+index);cl.setClear=false;}
else
    cl.displayLabel.setText(cl.displayLabel.getText()+index);
}//actionPerformed
}//class defination

/*********************************************/

class MyOperatorButton extends Button implements ActionListener
{
MyCalculator cl;
```

```java

MyOperatorButton(int x,int y, int width,int height,String cap, MyCalculator clc)
{
super(cap);
setBounds(x,y,width,height);
this.cl=clc;
this.cl.add(this);
addActionListener(this);
}
///////////////////////
public void actionPerformed(ActionEvent ev)
{
String opText=((MyOperatorButton)ev.getSource()).getLabel();

cl.setClear=true;
double temp=Double.parseDouble(cl.displayLabel.getText());

if(opText.equals("1/x"))
    {
    try
        {double tempd=1/(double)temp;
        cl.displayLabel.setText(MyCalculator.getFormattedText(tempd));}
    catch(ArithmeticException excp)
                        {cl.displayLabel.setText("Divide by 0.");}
    return;
    }
if(opText.equals("sqrt"))
    {
    try
        {double tempd=Math.sqrt(temp);
        cl.displayLabel.setText(MyCalculator.getFormattedText(tempd));}
            catch(ArithmeticException excp)
                    {cl.displayLabel.setText("Divide by 0.");}
    return;
    }
if(!opText.equals("="))
    {
    cl.number=temp;
    cl.op=opText.charAt(0);
    return;
    }
// process = button pressed
switch(cl.op)
{
case '+':
    temp+=cl.number;break;
case '-':
    temp=cl.number-temp;break;
case '*':
    temp*=cl.number;break;
case '%':
    try{temp=cl.number%temp;}
    catch(ArithmeticException excp)
        {cl.displayLabel.setText("Divide by 0."); return;}
    break;
case '/':
    try{temp=cl.number/temp;}
        catch(ArithmeticException excp)
                {cl.displayLabel.setText("Divide by 0."); return;}
    break;
}//switch

cl.displayLabel.setText(MyCalculator.getFormattedText(temp));
//cl.number=temp;
}//actionPerformed
}//class

/****************************************/

class MyMemoryButton extends Button implements ActionListener
{
```

```
MyCalculator cl;

/////////////////////////////////
MyMemoryButton(int x,int y, int width,int height,String cap, MyCalculator clc)
{
super(cap);
setBounds(x,y,width,height);
this.cl=clc;
this.cl.add(this);
addActionListener(this);
}
////////////////////////////////////////////////
public void actionPerformed(ActionEvent ev)
{
char memop=((MyMemoryButton)ev.getSource()).getLabel().charAt(1);

cl.setClear=true;
double temp=Double.parseDouble(cl.displayLabel.getText());

switch(memop)
{
case 'C':
    cl.memLabel.setText(" ");cl.memValue=0.0;break;
case 'R':
    cl.displayLabel.setText(MyCalculator.getFormattedText(cl.memValue));break;
case 'S':
    cl.memValue=0.0;
case '+':
    cl.memValue+=Double.parseDouble(cl.displayLabel.getText());
    if(cl.displayLabel.getText().equals("0") || cl.displayLabel.getText().equals("0.0")  )
        cl.memLabel.setText(" ");
    else
        cl.memLabel.setText("M");
    break;
}//switch
}//actionPerformed
}//class

/*******************************************/

class MySpecialButton extends Button implements ActionListener
{
MyCalculator cl;

MySpecialButton(int x,int y, int width,int height,String cap, MyCalculator clc)
{
super(cap);
setBounds(x,y,width,height);
this.cl=clc;
this.cl.add(this);
addActionListener(this);
}
//////////////////////
static String backSpace(String s)
{
String Res="";
for(int i=0; i<s.length()-1; i++) Res+=s.charAt(i);
return Res;
}

//////////////////////////////////////////////////////////
public void actionPerformed(ActionEvent ev)
{
String opText=((MySpecialButton)ev.getSource()).getLabel();
//check for backspace button
if(opText.equals("Backspc"))
{
String tempText=backSpace(cl.displayLabel.getText());
if(tempText.equals(""))
    cl.displayLabel.setText("0");
else
```

```
    cl.displayLabel.setText(tempText);
return;
}
//check for "C" button i.e. Reset
if(opText.equals("C"))
{
cl.number=0.0; cl.op=' '; cl.memValue=0.0;
cl.memLabel.setText(" ");
}

//it must be CE button pressed
cl.displayLabel.setText("0");cl.setClear=true;
}//actionPerformed
}//class

/*********************************************
Features not implemented and few bugs

i)  No coding done for "+/-" button.
ii) Menubar is not included.
iii)Not for Scientific calculation
iv)Some of the computation may lead to unexpected result
   due to the representation of Floating point numbers in computer
   is an approximation to the given value that can be stored
   physically in memory.
**********************************************/
```

download this example



Word Character Counter in Java with Source Code

Word Character Counter in Java with Source Code: We can develop Word Character Counter in java with the help of string, AWT/Swing with event handling. Let's see the code of creating Word Character Counter in java.

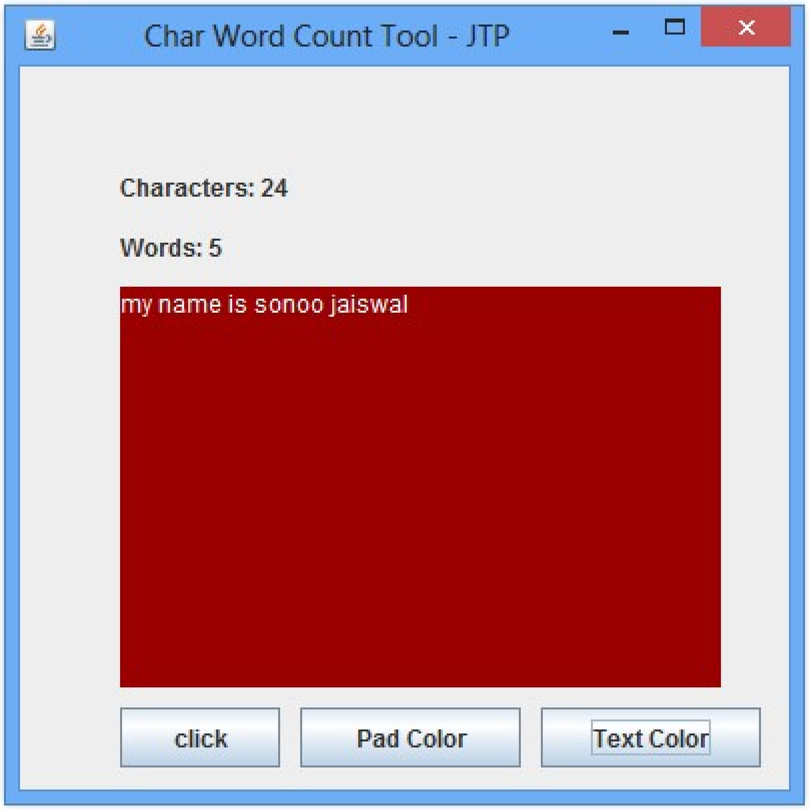
```
String text="hello javatpoint this is wcc tool";
String words[]=text.split("\\s");
int length=words.length;//returns total number of words
int clength=text.length();//returns total number of characters with space
```

Let's see the swing code to count word and character.

```java
import java.awt.event.*;
import javax.swing.*;
public class WCC extends JFrame implements ActionListener{
JTextArea ta;
JButton b1,b2;
WCC(){
    super("Word Character Counter - JavaTpoint");
    ta=new JTextArea();
    ta.setBounds(50,50,300,200);

    b1=new JButton("Word");
    b1.setBounds(50,300,100,30);

    b2=new JButton("Character");
    b2.setBounds(180,300,100,30);

    b1.addActionListener(this);
    b2.addActionListener(this);
    add(b1);add(b2);add(ta);
    setSize(400,400);
    setLayout(null);
    setVisible(true);
}
public void actionPerformed(ActionEvent e){
    String text=ta.getText();
    if(e.getSource()==b1){
        String words[]=text.split("\\s");
        JOptionPane.showMessageDialog(this,"Total words: "+words.length);
    }
    if(e.getSource()==b2){
        JOptionPane.showMessageDialog(this,"Total Characters with space: "+text.length());
    }
}
public static void main(String[] args) {
    new WCC();
}
}
```

Word Count Example with Pad and Text Color

```java
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
public class CharCount extends JFrame implements ActionListener{
    JLabel lb1,lb2;
    JTextArea ta;
    JButton b;
    JButton pad,text;
    CharCount(){
        super("Char Word Count Tool - JTP");
        lb1=new JLabel("Characters: ");
        lb1.setBounds(50,50,100,20);
        lb2=new JLabel("Words: ");
        lb2.setBounds(50,80,100,20);

        ta=new JTextArea();
        ta.setBounds(50,110,300,200);

        b=new JButton("click");
        b.setBounds(50,320, 80,30);//x,y,w,h
        b.addActionListener(this);

        pad=new JButton("Pad Color");
        pad.setBounds(140,320, 110,30);//x,y,w,h
        pad.addActionListener(this);

        text=new JButton("Text Color");
        text.setBounds(260,320, 110,30);//x,y,w,h
        text.addActionListener(this);

        add(lb1);add(lb2);add(ta);add(b);add(pad);add(text);

        setSize(400,400);
        setLayout(null);//using no layout manager
        setVisible(true);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }
    public void actionPerformed(ActionEvent e){
        if(e.getSource()==b){
        String text=ta.getText();
        lb1.setText("Characters: "+text.length());
        String words[]=text.split("\\s");
        lb2.setText("Words: "+words.length);
        }else if(e.getSource()==pad){
            Color c=JColorChooser.showDialog(this,"Choose Color",Color.BLACK);
            ta.setBackground(c);
        }else if(e.getSource()==text){
            Color c=JColorChooser.showDialog(this,"Choose Color",Color.BLACK);
            ta.setForeground(c);
        }
    }
public static void main(String[] args) {
    new CharCount();
}}
```



Characters: 24

Words: 5

my name is sonoo jaiswal

Online Exam Project in Java Swing without database

In this project, there are given 10 questions to play. User can bookmark any question for the reconsideration while going to result.

We are using here java array to store the questions, options and answers not database. You can use collection framework or database in place of array.

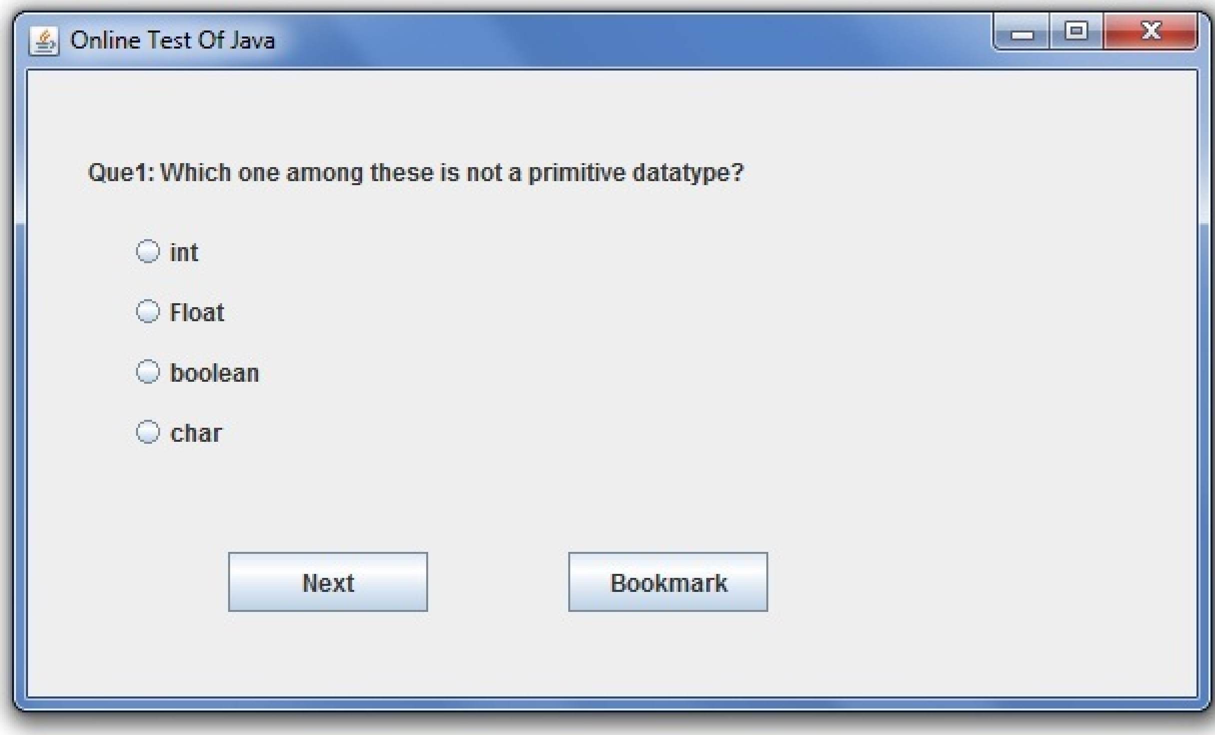
```
/*Online Java Paper Test*/

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class OnlineTest extends JFrame implements ActionListener
{
    JLabel l;
    JRadioButton jb[]=new JRadioButton[5];
    JButton b1,b2;
    ButtonGroup bg;
    int count=0,current=0,x=1,y=1,now=0;
    int m[]=new int[10];
    OnlineTest(String s)
    {
        super(s);
        l=new JLabel();
        add(l);
        bg=new ButtonGroup();
        for(int i=0;i<5;i++)
        {
            jb[i]=new JRadioButton();
            add(jb[i]);
            bg.add(jb[i]);
        }
        b1=new JButton("Next");
        b2=new JButton("Bookmark");
        b1.addActionListener(this);
        b2.addActionListener(this);
        add(b1);add(b2);
        set();
        l.setBounds(30,40,450,20);
        jb[0].setBounds(50,80,100,20);
        jb[1].setBounds(50,110,100,20);
        jb[2].setBounds(50,140,100,20);
        jb[3].setBounds(50,170,100,20);
        b1.setBounds(100,240,100,30);
        b2.setBounds(270,240,100,30);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(null);
        setLocation(250,100);
        setVisible(true);
        setSize(600,350);
    }
    public void actionPerformed(ActionEvent e)
    {
        if(e.getSource()==b1)
        {
            if(check())
                count=count+1;
            current++;
            set();
            if(current==9)
            {
                b1.setEnabled(false);
                b2.setText("Result");
            }
        }
        if(e.getActionCommand().equals("Bookmark"))
        {
            JButton bk=new JButton("Bookmark"+x);
            bk.setBounds(480,20+30*x,100,30);
            add(bk);
            bk.addActionListener(this);
            m[x]=current;
```

```java
                x++;
                current++;
                set();
                if(current==9)
                    b2.setText("Result");
                setVisible(false);
                setVisible(true);
            }
            for(int i=0,y=1;i<x;i++,y++)
            {
            if(e.getActionCommand().equals("Bookmark"+y))
            {
                if(check())
                    count=count+1;
                now=current;
                current=m[y];
                set();
                ((JButton)e.getSource()).setEnabled(false);
                current=now;
            }
            }

            if(e.getActionCommand().equals("Result"))
            {
                if(check())
                    count=count+1;
                current++;
                //System.out.println("correct ans="+count);
                JOptionPane.showMessageDialog(this,"correct ans="+count);
                System.exit(0);
            }
        }
        void set()
        {
            jb[4].setSelected(true);
            if(current==0)
            {
                l.setText("Que1: Which one among these is not a primitive datatype?");
                jb[0].setText("int");jb[1].setText("Float");jb[2].setText("boolean");jb[3].setText("char");
            }
            if(current==1)
            {
                l.setText("Que2: Which class is available to all the class automatically?");
                jb[0].setText("Swing");jb[1].setText("Applet");jb[2].setText("Object");jb[3].setText("ActionEvent");
            }
            if(current==2)
            {
                l.setText("Que3: Which package is directly available to our class without importing it?");
                jb[0].setText("swing");jb[1].setText("applet");jb[2].setText("net");jb[3].setText("lang");
            }
            if(current==3)
            {
                l.setText("Que4: String class is defined in which package?");
                jb[0].setText("lang");jb[1].setText("Swing");jb[2].setText("Applet");jb[3].setText("awt");
            }
            if(current==4)
            {
                l.setText("Que5: Which institute is best for java coaching?");
                jb[0].setText("Utek");jb[1].setText("Aptech");jb[2].setText("SSS IT");jb[3].setText("jtek");
            }
            if(current==5)
            {
                l.setText("Que6: Which one among these is not a keyword?");
                jb[0].setText("class");jb[1].setText("int");jb[2].setText("get");jb[3].setText("if");
            }
            if(current==6)
            {
                l.setText("Que7: Which one among these is not a class? ");
                jb[0].setText("Swing");jb[1].setText("Actionperformed");jb[2].setText("ActionEvent");
                        jb[3].setText("Button");
            }
            if(current==7)
            {
                l.setText("Que8: which one among these is not a function of Object class?");
                jb[0].setText("toString");jb[1].setText("finalize");jb[2].setText("equals");
                        jb[3].setText("getDocumentBase");
            }
            if(current==8)
            {
                l.setText("Que9: which function is not present in Applet class?");
                jb[0].setText("init");jb[1].setText("main");jb[2].setText("start");jb[3].setText("destroy");
            }
            if(current==9)
            {
                l.setText("Que10: Which one among these is not a valid component?");
                jb[0].setText("JButton");jb[1].setText("JList");jb[2].setText("JButtonGroup");
                        jb[3].setText("JTextArea");
            }
            l.setBounds(30,40,450,20);
            for(int i=0,j=0;i<=90;i+=30,j++)
                jb[j].setBounds(50,80+i,200,20);
        }
        boolean check()
        {
            if(current==0)
                return(jb[1].isSelected());
            if(current==1)
                return(jb[2].isSelected());
            if(current==2)
                return(jb[3].isSelected());
            if(current==3)
                return(jb[0].isSelected());
            if(current==4)
                return(jb[2].isSelected());
            if(current==5)
                return(jb[2].isSelected());
            if(current==6)
                return(jb[1].isSelected());
            if(current==7)
                return(jb[3].isSelected());
            if(current==8)
                return(jb[1].isSelected());
            if(current==9)
                return(jb[2].isSelected());
            return false;
        }
        public static void main(String s[])
        {
            new OnlineTest("Online Test Of Java");
        }
    }
```

download this mini project

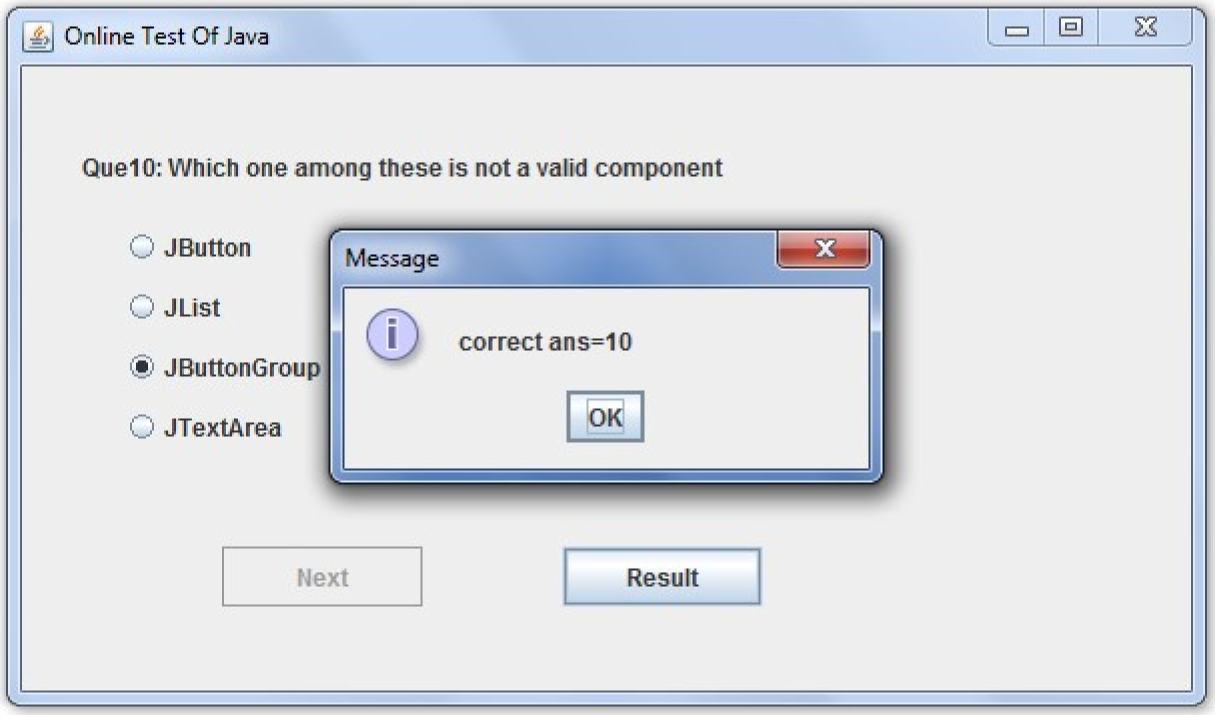
Output

Java Applet



Que1: Which one among these is not a primitive datatype?

- int
- Float
- boolean
- char



Que10: Which one among these is not a valid component

- JButton
- JList
- [✓ JButtonGroup]
- JTextArea

correct ans=10

Java Applet

Applet is a special type of program that is embedded in the webpage to generate the dynamic content. It runs inside the browser and works at client side.

Advantage of Applet

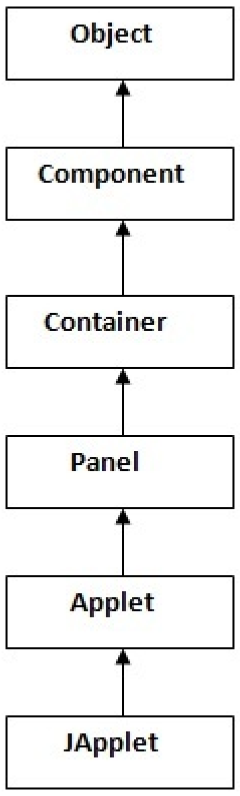
There are many advantages of applet. They are as follows:

- It works at client side so less response time.
- Secured
- It can be executed by browsers running under many plateforms, including Linux, Windows, Mac Os etc.

Drawback of Applet

- Plugin is required at client browser to execute applet.

Hierarchy of Applet



As displayed in the above diagram, Applet class extends Panel. Panel class extends Container which is the subclass of Component.

Lifecycle of Java Applet

1. Applet is initialized.
2. Applet is started.
3. Applet is painted.
4. Applet is stopped.
5. Applet is destroyed.

Lifecycle methods for Applet:

The java.applet.Applet class 4 life cycle methods and java.awt.Component class provides 1 life cycle methods for an applet.

java.applet.Applet class

For creating any applet java.applet.Applet class must be inherited. It provides 4 life cycle methods of applet.

1. **public void init():** is used to initialized the Applet. It is invoked only once.
2. **public void start():** is invoked after the init() method or browser is maximized. It is used to start the Applet.
3. **public void stop():** is used to stop the Applet. It is invoked when Applet is stop or browser is minimized.
4. **public void destroy():** is used to destroy the Applet. It is invoked only once.

java.awt.Component class

The Component class provides 1 life cycle method of applet.

1. **public void paint(Graphics g):** is used to paint the Applet. It provides Graphics class object that can be used for drawing oval, rectangle, arc etc.

Who is responsible to manage the life cycle of an applet?

Java Plug-in software.

How to run an Applet?

There are two ways to run an applet

1. By html file.
2. By appletViewer tool (for testing purpose).

Simple example of Applet by html file:

To execute the applet by html file, create an applet and compile it. After that create an html file and place the applet code in html file. Now click the html file.

```
//First.java
import java.applet.Applet;
import java.awt.Graphics;
public class First extends Applet{

public void paint(Graphics g){
g.drawString("welcome",150,150);
}

}
```

Note: class must be public because its object is created by Java Plugin software that resides on the browser.

myapplet.html

```
<html>
<body>
<applet code="First.class" width="300" height="300">
</applet>
</body>
</html>
```

Simple example of Applet by appletviewer tool:

To execute the applet by appletviewer tool, create an applet that contains applet tag in comment and compile it. After that run it by: appletviewer First.java. Now Html file is not required but it is for testing purpose only.

```
//First.java
import java.applet.Applet;
import java.awt.Graphics;
public class First extends Applet{

public void paint(Graphics g){
g.drawString("welcome to applet",150,150);
}

}
/*
<applet code="First.class" width="300" height="300">
</applet>
*/
```

To execute the applet by appletviewer tool, write in command prompt:

```
c:\>javac First.java
c:\>appletviewer First.java
```

Displaying Graphics in Applet

java.awt.Graphics class provides many methods for graphics programming.

Commonly used methods of Graphics class:

1. **public abstract void drawString(String str, int x, int y):** is used to draw the specified string.

2. **public void drawRect(int x, int y, int width, int height):** draws a rectangle with the specified width and height.
3. **public abstract void fillRect(int x, int y, int width, int height):** is used to fill rectangle with the default color and specified width and height.
4. **public abstract void drawOval(int x, int y, int width, int height):** is used to draw oval with the specified width and height.
5. **public abstract void fillOval(int x, int y, int width, int height):** is used to fill oval with the default color and specified width and height.
6. **public abstract void drawLine(int x1, int y1, int x2, int y2):** is used to draw line between the points(x1, y1) and (x2, y2).
7. **public abstract boolean drawImage(Image img, int x, int y, ImageObserver observer):** is used draw the specified image.
8. **public abstract void drawArc(int x, int y, int width, int height, int startAngle, int arcAngle):** is used draw a circular or elliptical arc.
9. **public abstract void fillArc(int x, int y, int width, int height, int startAngle, int arcAngle):** is used to fill a circular or elliptical arc.
10. **public abstract void setColor(Color c):** is used to set the graphics current color to the specified color.
11. **public abstract void setFont(Font font):** is used to set the graphics current font to the specified font.

Example of Graphics in applet:

```
import java.applet.Applet;
import java.awt.*;

public class GraphicsDemo extends Applet{

public void paint(Graphics g){
g.setColor(Color.red);
g.drawString("Welcome",50, 50);
g.drawLine(20,30,20,300);
g.drawRect(70,100,30,30);
g.fillRect(170,100,30,30);
g.drawOval(70,200,30,30);

g.setColor(Color.pink);
g.fillOval(170,200,30,30);
g.drawArc(90,150,30,30,30,270);
g.fillArc(270,150,30,30,0,180);

}
}
```

myapplet.html

```
<html>
<body>
<applet code="GraphicsDemo.class" width="300" height="300">
</applet>
</body>
</html>
```

Displaying Image in Applet

Applet is mostly used in games and animation. For this purpose image is required to be displayed. The java.awt.Graphics class provide a method drawImage() to display the image.

Syntax of drawImage() method:

1. **public abstract boolean drawImage(Image img, int x, int y, ImageObserver observer):** is used draw the specified image.

How to get the object of Image:

The java.applet.Applet class provides getImage() method that returns the object of Image. Syntax:

```
public Image getImage(URL u, String image){}
```

Other required methods of Applet class to display image:

1. **public URL getDocumentBase():** is used to return the URL of the document in which applet is embedded.
2. **public URL getCodeBase():** is used to return the base URL.

Example of displaying image in applet:

```
import java.awt.*;
import java.applet.*;


public class DisplayImage extends Applet {

  Image picture;

  public void init() {
    picture = getImage(getDocumentBase(),"sonoo.jpg");
  }

  public void paint(Graphics g) {
    g.drawImage(picture, 30,30, this);
  }

  }
```

In the above example, drawImage() method of Graphics class is used to display the image. The 4th argument of drawImage() method of is ImageObserver object. The Component class implements ImageObserver interface. So current class object would also be treated as ImageObserver because Applet class indirectly extends the Component class.

myapplet.html

```
<html>
<body>
<applet code="DisplayImage.class" width="300" height="300">
</applet>
</body>
</html>
```

Animation in Applet

Example of animation in applet:

```
import java.awt.*;
import java.applet.*;
public class AnimationExample extends Applet {

  Image picture;

  public void init() {
    picture =getImage(getDocumentBase(),"bike_1.gif");
  }
```

```
    public void paint(Graphics g) {
      for(int i=0;i<500;i++){
        g.drawImage(picture, i,30, this);

        try{Thread.sleep(100);}catch(Exception e){}
      }
    }
  }
```

In the above example, drawImage() method of Graphics class is used to display the image. The 4th argument of drawImage() method of is ImageObserver object. The Component class implements ImageObserver interface. So current class object would also be treated as ImageObserver because Applet class indirectly extends the Component class.

myapplet.html

```
<html>
<body>
<applet code="DisplayImage.class" width="300" height="300">
</applet>
</body>
</html>
```

7. EventHandling in Applet

As we perform event handling in AWT or Swing, we can perform it in applet also. Let's see the simple example of event handling in applet that prints a message by click on the button.

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
public class EventApplet extends Applet implements ActionListener{
Button b;
TextField tf;

public void init(){
tf=new TextField();
tf.setBounds(30,40,150,20);

b=new Button("Click");
b.setBounds(80,150,60,50);

add(b);add(tf);
b.addActionListener(this);

setLayout(null);
}

 public void actionPerformed(ActionEvent e){
  tf.setText("Welcome");
 }
}
```

In the above example, we have created all the controls in init() method because it is invoked only once.

myapplet.html

```
<html>
<body>
<applet code="EventApplet.class" width="300" height="300">
</applet>
</body>  </html>
```

JApplet class in Applet

As we prefer Swing to AWT. Now we can use JApplet that can have all the controls of swing. The JApplet class extends the Applet class.

```
import java.applet.*;
import javax.swing.*;
import java.awt.event.*;
public class EventJApplet extends JApplet implements ActionListener{
JButton b;
JTextField tf;
public void init(){

tf=new JTextField();
tf.setBounds(30,40,150,20);

b=new JButton("Click");
b.setBounds(80,150,70,40);

add(b);add(tf);
b.addActionListener(this);

setLayout(null);
}

public void actionPerformed(ActionEvent e){
tf.setText("Welcome");
}
}
```

In the above example, we have created all the controls in init() method because it is invoked only once.

myapplet.html

```
<html>
<body>
<applet code="EventJApplet.class" width="300" height="300">
</applet>
</body>
</html>
```

Painting in Applet

We can perform painting operation in applet by the mouseDragged() method of MouseMotionListener.

```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
public class MouseDrag extends Applet implements MouseMotionListener{

public void init(){
addMouseMotionListener(this);
setBackground(Color.red);
}

public void mouseDragged(MouseEvent me){
Graphics g=getGraphics();
g.setColor(Color.white);
g.fillOval(me.getX(),me.getY(),5,5);
}
public void mouseMoved(MouseEvent me){}

}
```

In the above example, getX() and getY() method of MouseEvent is used to get the current x-axis and y-axis. The getGraphics() method of Component class returns the object of Graphics.

myapplet.html

```
<html>
<body>
<applet code="MouseDrag.class" width="300" height="300">
</applet>
</body>
</html>
```

Digital clock in Applet

Digital clock can be created by using the Calendar and SimpleDateFormat class. Let's see the simple example:

```
import java.applet.*;
import java.awt.*;
import java.util.*;
import java.text.*;

public class DigitalClock extends Applet implements Runnable {

  Thread t = null;
  int hours=0, minutes=0, seconds=0;
  String timeString = "";

  public void init() {
    setBackground( Color.green);
  }

  public void start() {
       t = new Thread( this );
       t.start();
  }


  public void run() {
    try {
      while (true) {

        Calendar cal = Calendar.getInstance();
        hours = cal.get( Calendar.HOUR_OF_DAY );
        if ( hours > 12 ) hours -= 12;
        minutes = cal.get( Calendar.MINUTE );
        seconds = cal.get( Calendar.SECOND );

        SimpleDateFormat formatter = new SimpleDateFormat("hh:mm:ss");
        Date date = cal.getTime();
        timeString = formatter.format( date );

        repaint();
        t.sleep( 1000 );  // interval given in milliseconds
      }
    }
    catch (Exception e) { }
  }


 public void paint( Graphics g ) {
    g.setColor( Color.blue );
    g.drawString( timeString, 50, 50 );
  }
}
```

In the above example, getX() and getY() method of MouseEvent is used to get the current x-axis and y-axis. The getGraphics() method of Component class returns the object of Graphics.

myapplet.html

```
<html>
<body>
<applet code="DigitalClock.class" width="300" height="300">
</applet>
</body>
</html>
```

Analog clock in Applet

Analog clock can be created by using the Math class. Let's see the simple example:

Example of Analog clock in Applet:

```
import java.applet.*;
import java.awt.*;
import java.util.*;
import java.text.*;

public class MyClock extends Applet implements Runnable {

   int width, height;
   Thread t = null;
   boolean threadSuspended;
   int hours=0, minutes=0, seconds=0;
   String timeString = "";

   public void init() {
      width = getSize().width;
      height = getSize().height;
      setBackground( Color.black );
   }

   public void start() {
      if ( t == null ) {
         t = new Thread( this );
         t.setPriority( Thread.MIN_PRIORITY );
         threadSuspended = false;
         t.start();
      }
      else {
         if ( threadSuspended ) {
            threadSuspended = false;
            synchronized( this ) {
               notify();
            }
         }
      }
   }

   public void stop() {
      threadSuspended = true;
   }

   public void run() {
      try {
         while (true) {

            Calendar cal = Calendar.getInstance();
            hours = cal.get( Calendar.HOUR_OF_DAY );
            if ( hours > 12 ) hours -= 12;
```

```
               minutes = cal.get( Calendar.MINUTE );
               seconds = cal.get( Calendar.SECOND );

               SimpleDateFormat formatter
                  = new SimpleDateFormat( "hh:mm:ss", Locale.getDefault() );
               Date date = cal.getTime();
               timeString = formatter.format( date );

               // Now the thread checks to see if it should suspend itself
               if ( threadSuspended ) {
                  synchronized( this ) {
                     while ( threadSuspended ) {
                        wait();
                     }
                  }
               }
               repaint();
               t.sleep( 1000 );  // interval specified in milliseconds
            }
         }
         catch (Exception e) { }
      }

      void drawHand( double angle, int radius, Graphics g ) {
         angle -= 0.5 * Math.PI;
         int x = (int)( radius*Math.cos(angle) );
         int y = (int)( radius*Math.sin(angle) );
         g.drawLine( width/2, height/2, width/2 + x, height/2 + y );
      }

      void drawWedge( double angle, int radius, Graphics g ) {
         angle -= 0.5 * Math.PI;
         int x = (int)( radius*Math.cos(angle) );
         int y = (int)( radius*Math.sin(angle) );
         angle += 2*Math.PI/3;
         int x2 = (int)( 5*Math.cos(angle) );
         int y2 = (int)( 5*Math.sin(angle) );
         angle += 2*Math.PI/3;
         int x3 = (int)( 5*Math.cos(angle) );
         int y3 = (int)( 5*Math.sin(angle) );
         g.drawLine( width/2+x2, height/2+y2, width/2 + x, height/2 + y );
         g.drawLine( width/2+x3, height/2+y3, width/2 + x, height/2 + y );
         g.drawLine( width/2+x2, height/2+y2, width/2 + x3, height/2 + y3 );
      }

      public void paint( Graphics g ) {
         g.setColor( Color.gray );
         drawWedge( 2*Math.PI * hours / 12, width/5, g );
         drawWedge( 2*Math.PI * minutes / 60, width/3, g );
         drawHand( 2*Math.PI * seconds / 60, width/2, g );
         g.setColor( Color.white );
         g.drawString( timeString, 10, height-10 );
      }
   }
```

myapplet.html

```
<html>
<body>
<applet code="MyClock.class" width="300" height="300">
</applet>
</body>
</html>
```

Parameter in Applet

We can get any information from the HTML file as a parameter. For this purpose, Applet class provides a method named getParameter().

Syntax:

```
public String getParameter(String parameterName)
```

```
import java.applet.Applet;
import java.awt.Graphics;

public class UseParam extends Applet{

public void paint(Graphics g){
String str=getParameter("msg");
g.drawString(str,50, 50);
}

}
```

myapplet.html

```
<html>
<body>
<applet code="UseParam.class" width="300" height="300">
<param name="msg" value="Welcome to applet">
</applet>
</body>
</html>
```